DOCUMENT RESUME

ED 243 651 SE 044 382

AUTHOR Beam, John; And Others
TITLE Computer Mathematics 1A. Curriculum Bulletin Number 82CBM32.
INSTITUTION Houston Independent School District, Tex.
PUB DATE 82
NOTE 233p.
PUB TYPE Guides - Classroom Use - Guides (For Teachers) (052)

EDRS PRICE MF01/PC10 Plus Postage.
DESCRIPTORS *Computer Science Education; Computer Software; Curriculum Guides; Educational Objectives; Learning Activities; *Mathematics Curriculum; *Mathematics Instruction; *Programing; Resource Materials; Secondary Education; *Secondary School Mathematics; Teaching Methods
IDENTIFIERS *BASIC Programing Language; Texas (Houston)

ABSTRACT

This guide for teachers in the Houston schools outlines the first semester of a two-semester course on computer mathematics. Course objectives are listed, and a scope-and-sequence chart correlates topics in the course with the textbook ("Computer Programming in the BASIC Language") and suggests time allotments. An objectives-by-topic cross-index is also included, and supplements are indexed to topics. The major portion of the guide presents plans for twelve topics aimed at teaching students how to program the computer for mathimatical exercises using the BASIC programing language. Each topic lists objectives, with textbook references, teaching suggestions, and student activities for each objective. Appended are computer programs, plus lists of bibliographic references, films, and filmstrips. (MNS)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED243651

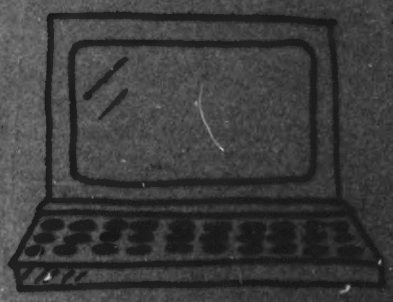
U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

✓ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

COMPUTER MATHEMATICS 6



see the computer

see the computer run

run, computer, run!

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY
Sarah Abbott

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

SE044382

Houston Independent School District

Curriculum Bulletin Number 33[illegible]

Item Number 33.0390

COMPUTER MATHEMATICS 1A

Curriculum Bulletin Number 82CBM32

Prepared by

John Beam, Bellaire High School
Carolyn Eubanks, Worthing High School
Elisabeth Jones, Sharpstown High School

Published by

Houston Independent School District
Houston, Texas
1982

Prepared Under the Direction of

Ms. Madolyn Reed, Director, Mathematics Department
Mrs. Bettye Hall, Assistant Director, Secondary Mathematics
Mrs. Roberta Brisco, Assistant Director, Curriculum Publications

Board of Education

Mr. Ray Morrison, President
Mr. Tarrant Fendley, Vice-President
Mrs. Cathy Mincberg, Secretary
Mrs. Marquis L. (Blue) Alexander, Jr., Assistant Secretary
Mr. Wiley Henry
Dr. James C. Jones
Mrs. Bobby Ann Peiffer
Ms. Tina Reyes
Mrs. Elizabeth Spates

Administration

Mr. Billy R. Reagan, General Superintendent
Mrs. Patricia Shell, Superintendent for Instruction
Mr. Lawrence Marshall, Deputy Superintendent, General Instructional Services and Alternative Education

PREFACE

It is our intention that this curriculum will benefit all computer teachers. Hopefully, whether you are a first-year teacher or an "old-pro" there will be ideas to encourage and challenge you in your endeavor.

A few special notes to those of you teaching computer math for the first time:

(1) Check your school library and audiovisual department at the beginning of the year for resource materials immediately available to you.

(2) Class load should be dependent on equipment. Most schools do not have enough equipment to allow students to work independently. After the students have all had a chance to familiarize themselves with the equipment (see ELO 7) and classmates, allow them to sign up for programming groups. Stress that each student must contribute to the group in order to master the course objectives. Reserve the right to regroup students to achieve a balance of participation. Learning to work in groups is a beneficial experience for students because project teams are used in almost every phase of data processing. Groups should have a maximum of four members; two or three member groups would be preferable. Each group will have a specific amount of computer time (other groups will help monitor a group's time). As a general rule, each machine with printout and/or disk capabilities should be able to handle three groups of three members each (nine students) during a one-hour class period.

(3) Students should meet minimum requirements as set up by H.I.S.D. They must have completed one year of math (no remedial courses are counted).

(4) An overhead projector will aid you greatly in presentation of sample programs and instructions.

(5) Supplementary material will be necessary to cover the course objectives in the guide.

(6) Students should be encouraged to keep notebooks. These may prove useful to them in the future in college courses and/or jobs.

(7) Field trips and/or guest speakers are desirable. Students need to see large-scale data processing. Oil companies, hospitals, and manufacturers of data processing equipment are usually willing to schedule tours, but call well in advance. Wait until students have become familiar with computer terminology before planning a trip.

(8) At least 50 percent of a student's grade should be based on individual testing or projects. Group computer projects and daily exercise would comprise the other portion of a student's grade.

(9) One final comment must be made about the programs in the appendix to this guide. In many cases an objective poses a problem which is answered by reference to a BASIC program already in the text. In these cases the book's program is usually adequate to satisfy the objective. However, the appendix may also contain one or more programs that solve essentially the same problem, either by an alternative method or by one which is more efficient or elegant than that in the text. Such programs are included as reference for the more advanced, eager student or the teacher who may wish to examine alternatives and obtain ideas about programming efficiency and style. However, they should not be used as standards by which to judge the class. To do so might well prove discouraging both to the students and to the teacher. Finally, some programs, for example matrix inversion for an arbitrary N x N matrix, are included for the possible later use of the teacher or student. However, it is not expected or necessarily desirable that the student or teacher trace through the programming logic in detail.

Table of Contents

	Page
Preface	iii
Course Objectives	1
Scope and Sequence	5
Objectives-to-Topics Cross Index	7
List of Supplements, Indexed to Topics	8
Topics	11
Topic One: People and Computers	13
Topic Two: Elements of Basic	45
Topic Three: Data In and Data Out	79
Three Four: Flowcharts and Program Branching	87
Topic Five: For-Next Loops	109
Topic Six: Accumulating and Table Heading	119
Topic Seven: Basic Functions	129
Topic Eight: Subscripting	141
Topic Nine: Nested Loops and Sorting	151
Topic Ten: Alphameric Output and Input	165
Topic Eleven: Two-Dimensional Arrays	175
Topic Twelve: Functions and Subroutines	183
Appendix	213
Program Appendix	215
HISD Instructional Media Services	231
Catalog of 16mm Computer Films	234

COMPUTER MATHEMATICS A

COURSE OBJECTIVES

By the end of the first semester the student will:

1. Demonstrate computer literacy by being able to recognize and use in proper context the vocabulary designated in the curriculum guide.*
2. Trace the historical development of the computer.*
3. Draw a block diagram of the computer system, label the essential parts and describe three functions of the CPU, and categorize input/output devices and media.*
4. List and describe computer-related careers, stating educational and experience requirements, job opportunities, and current salary ranges.*
5. Write a paper on a computer-related topic.*
6. Flowchart simple problems using standard symbols.
7. Use the system commands necessary to enter and run a program.
8. Write BASIC programs from flowcharts containing terminal and assignment symbols.
9. Make a tract of a BASIC program with ten or more statements, in order to determine its output.
10. Debug a BASIC program with ten or more statements.
11. Compute the value of arithmetic statements involving addition, subtraction, multiplication, division, exponentiation, and parentheses according to the predefined hierarchy of operations.
12. Write and flowchart BASIC programs which utilize the following: READ... DATA, INPUT, PRINT, LET, GOTO, IF...THEN..., FOR...NEXT, STOP, nested FOR, ...NEXT, END, REM, RESTORE, arithmetic operators, and relational operators.
13. Write BASIC programs using available library functions including RND(X), INT(X), ABS(X), and SQR(X).

13a. Write BASIC programs using available library functions including SIN(X), COS(X), ATN(X), LOG(X), and EXP(X).

*It is important for students to gain actual programming experience as soon as possible. ELO's 1-5 should be implemented throughout the semester, but ELO's beginning with 6 should be undertaken at once. If students are to take both semesters, ELO's 4 and 5 can be deferred until the second semester.

(14.) Write BASIC programs which utilize single subscripts, string variables, and the TAB function.

14a. Write BASIC programs which utilize double subscripts, ON...GOTO, and the functions SPC, CLEAR, PRINT USING, SET, and RESET.

(15.) Write a BASIC program appropriate for the highest level mathematics course the student has taken.

(16.) Do a completely documented term project: a heading (name, date, program title), statement of the problem, description of method, flowchart, listing of the program, the output of the program, and an evaluation (discussion and analysis) of the program.

(17.) By the end of the first semester the student should have sufficient programming experience that he will be able to write BASIC programs that will -

(a) Take the odd integers from one to twenty, and print the number, its square, and its cube, printing the values in columns with the appropriate headings.

(b) Calculate n! for n = 1 to 15 and print out n and n! in appropriately headed columns.

(c) Generate prime numbers less than 300 and list across the page.

(d) List the integral divisors of any integer less than 100, when the integer data are entered through an INPUT statement.

(e) Select and print out the largest and smallest numbers in a set of data.

(f) Accept any integer with a maximum of six digits and print out the value of each digit and the value of the position it occupies.

(g) Generate n terms of the Fibonacci series for $n < 25$ and find their sum.

(h) Print all the prime factors of a given integer less than 500.

(i) Produce n distinct random integers for $n \leq 50$.

(j) Accept three integers m, n, and k, and randomly generate k integers between m and n.

(k) Manipulate numerical data and round the results to a given number of decimal places.

(l) Use coordinate geometry formulas to make decisions about geometric figures.

(m) Arrange a set of numbers in order from least to greatest.

(n) Alphabetize a list of words.

(o) Plot the curve of a simple function.

17a. By end of the first semester the more advanced student should also be able to write programs that will -

(a) Find the greatest common factor and least common multiple of any two positive integers by using the Euclidean algorithm.

(b) Determine the ten smallest Ramanujan numbers (those which can be expressed as the sum of two cubes in more than one way).

(c) Find both solutions of a quadratic equation, whether real or complex.

(d) Convert Arabic numerals to Roman numerals.

(e) Print an NxN array of numbers and check whether it is either a magic square or a Latin square.

(f) Calculate, given n and k, the binomial coefficient $\binom{n}{k} = C^{n}_{k}$.

(g) Play tic-tac-toe, allowing either the student or the computer to start. (This is a lengthy problem, and could be used as a project.)

Vocabulary for Computer Mathematics A

Address, argument, command, system command, list, run, execute, input, output, on line, off line
Algorithm, code, ASCII, EBCDIC, program, statement, assembly language, machine language, high-level language
Alphanumeric, numeric, array, integer, real number, string, subscript, variable, floating point
Analog computer, digital computer, microcomputer, minicomputer, terminal, chip, core memory, CPU, CRT, RAM, ROM
Batch operating system, multiprocessor, real time, real-time system, time-sharing system
Binary, octal, decimal, hexadecimal, bit, byte, word
Branching, multiple branching, index, initialization, loop, nested loop
Bug, debugging, editing, GIGO (garbage in, garbage out)
Counter, cursor
Data, data processing, format, syntax
Documentation, flowchart, heading
First generation computer, second generation computer, third generation computer
Floppy disk, hard disk, hard copy, hardware, software, magnetic tape, modem, user
Millisecond, microsecond, nanosecond

COMPUTER MATHEMATICS

SCOPE AND SEQUENCE

PREREQUISITE: One non-remedial credit in mathematics

CREDIT: 1/2, 1 Credit

TEXTBOOK: Computer Programming in the BASIC Language
Jacobs, French, Moulds, and Schuchman
Allyn and Bacon, Inc.

AVERAGE COURSE SCOPE AND SEQUENCE*	TEXTBOOK CORRELATION	SUGGESTED TIME ALLOTMENT*
COMPUTER MATHEMATICS A		
People and Computers	Topic 1	5 days
Elements of Basic	Topic 2	5 days
Data In and Out	Topic 3	5 days
Flow Charts and Program Branching	Topic 4	10 days
For-Next Loops	Topic 5	10 days
Accumulating and Table Heading	Topic 6	5 days
BASIC Functions	Topic 7	10 days
Subscripting	Topic 8	10 days
Nested Loops and Sorting	Topic 9	12 days
Alphameric Output and Input	Topic 10	5 days
Term Project**	Obj. A16	8 days
Review and Final		2 days
		87 days

* and ** See bottom of next page.

AVERAGE COURSE SCOPE AND SEQUENCE*	TEXTBOOK CORRELATION	SUGGESTED TIME ALLOTMENT*
COMPUTER MATHEMATICS B		
Two-Dimensional Arrays	Topic 11	10 days
Functions and Subroutines	Topic 12	10 days
Matrix Algebra	Topic 13	10 days
Converging Processes	Topic 14	10 days
Alphabetic Sort and Curve Plotting (Obj. A17(n), A17(o) Already in Topics 10, 12)	Appendix	5 days
Optional Objectives	None	20 days
Term Project**	Obj. 12b	15 days
Review and Final		2 days
		82 days

*Computer Mathematics is best taught through flexible scheduling of class time. Topic I can be spread throughout the course. Some teachers may wish to begin flow charting immediately. The pace a teacher sets depends not only upon the preparation and aptitude of the students but also upon the number of students in the class and the number of computer terminals available. Some classes require two semesters to complete the 14 topics and others may finish in one semester, and can devote more time to the optional objectives.

**When assigning term projects a teacher must take into account the mathematical background and maturity of the individual student. A wide variety of topics should be presented from which the student may choose a single topic for a project.

Objectives-to-Topics Cross Index

Objective	Location Where Discussed
A1	All topics (implicitly) Additional Obj. A.B4
A2	Topic 1
A3	Topic 1
A4	Topic 1
A5	Topic 1
A6	Topic 4
A7	Topic 1
A8	Topic 4
A9	Topics 2, 4
A10	Topic 1, 5
A11	Topic 2
A12	Topics 2, 3, 4, 5, 9
A13	Topic 7
A13a	Topic 7
A14	Topics 6, 8, 10
A14a	Topics 5, 10, 11, Additional Obj. B8a
A15	Additional Objectives - A15
A16	Topic 4
A17(a)	Topic 6
A17(b)	Topic 8, Program Appendix
A17(c)	Topic 12, Program Appendix
A17(d)	Topic 12, Program Appendix
A17(e)	Topic 9
A17(f)	Topic 8, Program Appendix
A17(g)	Topic 8, Program Appendix
A17(h)	Topic 12, Program Appendix
A17(i)	Topic 7
A17(j)	Topic 7
A17(k)	Topic 7, Program Appendix
A17(l)	Topic 7
A17(m)	Topic 9
A17(n)	Topic 10
A17(o)	Topic 1a2
A17a(a)	Additional Obj. - A, Prog. Ap.
A17a(b)	Additional Obj. - A, Prog. Ap.
A17a(c)	Additional Obj. - A, Prog. Ap.
A17a(d)	Additional Obj. - A, Prog. Ap.
A17a(e)	Topic 11, Program Appendix
A17a(f)	Additional Obj. - A, Prog. Ap.
A17a(g)	Additional Obj. - A

List of Supplements, Indexed to Topics

Topic Supplement Title (Each supplement is coded by the corresponding objective number.)

1 People and Computers (Objectives A2, A3, A4, A5, A7, A10)

A2.1	A History of the Computer
A2.2	Computer Scientists Activity
A3.1	Block Diagram of a Computer System
A3.2	Input/Output Media Activity
A3.3	Storing Information in a Computer
A4	Occupations Project
A5	You and Computers
A7.1	Operator Test
A7.2	Region IV Log-on/Log-off Procedures
A7.3	Commands Used with Region IV
A7.4	Topic One Test (Test I)

2 Elements of BASIC (Objectives A9, A11, A12)

A11	Hierarchy Problems
A12.1	Basic BASIC (15 p. document by Dorothy Leach)
A12.2	Punching Paper Tape
A12.3	Entering and Running Programs from Paper Tape

3 Data In and Data Out (Objective A12)

A12.4	System Spacing Test for Students

4 Flow Charts and Program Branching (Objectives A6, A8, A9, A12, A16)

A6.1	Flowcharting Exercises I
A6.2	Flowcharting Exercises II
A9.1	Tracing a Program
A9.2	Trace Sheet
A16.1	Program Documentation
A16.2	Sample Documented Program

5 For-Next Loops (Objectives A10, A12)

A12.4	System Spacing Test for Students

6 Accumulating and Table Heading (Objectives A14, A14a, A14(a))

A14.1	Computer Designs

7 BASIC Functions (Objectives A13, A13a, A17(i), A17(j), A17(k), A17(l)

A17(l)	Coordinate Geometry

8 Subscripting (Objectives A14, A17(b), A17(f), A17(g))

A17(g)	Extended Fibonacci Sequences

Topic Supplement Title, (Each supplement is coded by the corresponding objective number.)

9 Nested loops and Sorting (Objectives A12, A17(e), A17(m))

A12.6 Parkside's Triangle

10 Alphametic Output and Input (Objectives A14, A14a, A17(n))

A14.2 Review Test

Additional Objectives - A (Objectives A15, A17a(a), A17a(b), A17a(c), A17a(d), A17a(f), A17a(g)

A15 Problems Appropriate for Various Levels of Mathematics
A17a(a) Euclidean algorithm
A17a(c) Roots of a quadratic equation (student correction exercise)
A17a(d) Arabic to Roman Numerals (Program)

11 Two Dimensional Arrays (Objectives A14a, B6, A17a(e))
None

12 Functions and Subroutines (Objectives A17(c), A17(d), A17(h), A17(o), B5)

B5 Program: Graphics of Three-Dimensional Functions

TOPICS

TOPIC ONE: PEOPLE AND COMPUTERS

Performance Objectives:

The student will -

A2 - Trace the historical development of the computer.

A3 - Draw a block diagram of a computer system and label the essential parts; describe three functions of the CPU (arithmetic, logic, control); categorize input/output devices and media.

A4 - List and describe computer-related careers, stating educational and experience requirements, job opportunities, and current salary ranges.

A5 - Write a paper on a computer-related topic.

A7 - Use system commands necessary to enter and run a program.

A10 - Debug a BASIC program of ten or more statements.

References:

Textbook pages 1-35

Objective A2: The student will trace the historical development of the computer.

Textbook Reference: Pp. 2-3 The Development of Calculating Machines
Pp. 106-107 Charles Babbage
Pp. 340-341 Herman Hollerith
Pp. 360-361 von Neumann

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
The history of the computer can be presented by lecture, film, handout, class reports, or a combination of any of these methods. One of the most effective, succinct methods would be the presentation of a film, followed by a handout of Supplement 2.1. An alternative would be using Supplement 2.2 as a guide to student note-taking while viewing a film or listening to a lecture. The student should be tested on this material on an individual basis. This objective can be covered adequately in one to two days. It may be expanded, however, according to teacher choice, student interest, and available class time. References: (Filmstrip and cassette) The History of the Computer. Chicago, Illinois: Encyclopedia Britannica Educational Corporation, 1972, from the set How Computers Work.	Supplement A2.1 Supplement A2.2

A History of the Computer

As early as 3000 B.C. man used the abacus to aid in counting; the abacus was the forerunner of the digital computer* - the beads were the "digits" and the wires were the places or "addresses." The first analog computer was developed in the early 1600's by Napier; his calculating rods were the forerunner of the modern slide rule. William Oughtred actually developed the slide rule based on Napier's calculations.

The first manual digital computer, an adding machine, was developed in 1642 by Blaise Pascal, a brilliant mathematician, who wanted to ease his father's clerking duties in the tax office. His machine was very similar to modern machines, featuring wheels and a display area. Of course, it was mechanical, not electrical. In 1671 Leibnitz developed a machine called the Stepped Reckoner that could also multiply. However, the design was not successfully executed until 1694, and it lacked accuracy. The trigonometry and logarithm tables had not been developed, and some mathematicians spent their whole lives calculating on these tables. These machines of the 1700's were, therefore, an improvement over nonmechanical methods.

The next significant person in computer history did not emerge until the beginning of the 19th century. Charles Babbage is often called "the father of the computer." He was from a wealthy English family, attended Cambridge, and devoted his life to the computer. In 1882 he constructed the "Difference Engine." His imagination was far ahead of the engineering techniques of the day. His second machine, the "Analytical Engine," was never completed despite generous government grants; this was mainly because he continually changed the plans, thus delaying actual construction. His most enthusiastic supporter, Lady Lovelace, a daughter of Lord Byron, worked with him on a scheme to pick winners at the race track with his machine. Babbage was an interesting, yet tragic, figure because he was so far ahead of his times. It was not until 1940 that anyone built a machine to compare with his designs.

In the early 1800's, Joseph Jacquard used perforated cards to control weaving loom operations. Following the ideas of the Frenchman, Herman Hollerith was the first American to advance the techniques of computing. His punched paper tapes and cards used in taking the 1890 census led him into the business of data processing. His company merged with three others to form the firm that was to become IBM.

Americans again were in the forefront of computer development with the advent of World War II. In 1944, Mark I was put into operation at Harvard by Dr. Howard Aiken - it was the first electromechanical digital computer. The first fully electronic computer, ENIAC, was developed at the University of Pennsylvania by J. Presper Eckert and John Mauchly; with its vacuum tubes, it was the largest computer, covering 15,000 square feet. John von Neumann, a noted mathematician,

---

*In the analog computer, numbers are represented by some continuously variable quantity, such as length, volume, or voltage level. Analog calculators and computers are limited in their accuracy by the precision with which measurements of such quantities can be made, whereas in a digital computer, there is no measuring, only counting.

originated the stored-program concept in the mid-40's. The ENIAC was modified, and eventually EDSAC abd EDVAC evolved to implement Neumann's idea. UNIVAC was the first computer to sell commercially. This was in the early 1950's. The second generation computers (late 1950's) decreased somewhat in physical size by the use of transistors, and the third and present generation (mid 1960's) enables the development of very small units through microcircuitry.

Different programming languages evolved during the 50's and 60's. In 1955 FORTRAN (FORmula TRANslation) was developed as a scientific programming language. In the early 1960's COBOL (COmmon Business Oriented Language) was designed as a business language. Many other languages have been developed, such as ALGOL, APL, PASCAL, and PLI, but FORTRAN, COBOL, and BASIC remain in the forefront. The development of database management systems such as ADA, Inquire, System 2000 and IMS will undoubtedly affect the future of data processing; they all feature a set of programs which manipulate the data.

There is a current trend toward decentralization of computer facilities. As of now, there is no recognized fourth generation of computers, but computers continue to spread their influence in every phase of human life. Computerized shopping, environment control, and human memory expansion are only a few of the areas of future development.

Computer Scientists Activity

Name	Date	Contribution:
John Napier	1614	
William Oughtred	1622	
Blaise Pascal	1642	
Gottfried von Leibnitz	1671	
Joseph Jacquard	1801	
Charles Babbage	1850	
Herman Hollerith	1887	
Howard Aiken	1944	
Mauchly and Eckert	1946	
J. von Neumann	1946	

Objective A3: The student will draw a block diagram of a computer system and label the essential parts; describe three functions of the CPU (Arithmetic, Logic, and Control); categorize input/output devices and media.

Textbook Reference: Pp. 10-11 - The Computer Processing Cycle

TEACHING SUGGESTIONS

Many introductory data processing books include a discussion and/or diagram of the computer processing cycle. Supplement 3.1 describes the process in more general terms than the diagram on page 11 in the textbook; it could be presented on the board, on the overhead projector, or as a handout. The students may be asked to reproduce the diagram on a test. Allow some variation in student drawings, provided they show a general understanding of the process. Key words, such as input and output, cannot be omitted, however.

Along with discussion of the diagram, the various modes of input and output should be discussed. Point out advantages and disadvantages of various methods. Ask students to describe devices and media they have observed. Don't limit the students' knowledge to classroom facilities. A film could give graphic illustrations of the various types of computer equipment. Students may be given a handout (Supplement 3.2) and asked to fill it out during or after a film presentation. Students may be asked to recall a few of the items in each category on a test. A field trip will further enforce the concepts in this objective (see preface).

References:

Bent, Robert J. and Sethares, George C. Business BASIC. Monterey, California: Brooks/Cole Publishing Company, 1980, pp. 1-5.

Golden, Neal. Computer Programming in the BASIC Language. New York City, New York: Harcourt Brace Jovanovich, 1975, p. 2.

Poirot, James and Grove, David. Computers and Mathematics. Manchaca, Texas: Sterling Swift Publishing Company, 1979, pp. 16-60.

STUDENT ACTIVITIES IN THIS GUIDE

Supplement A3.1

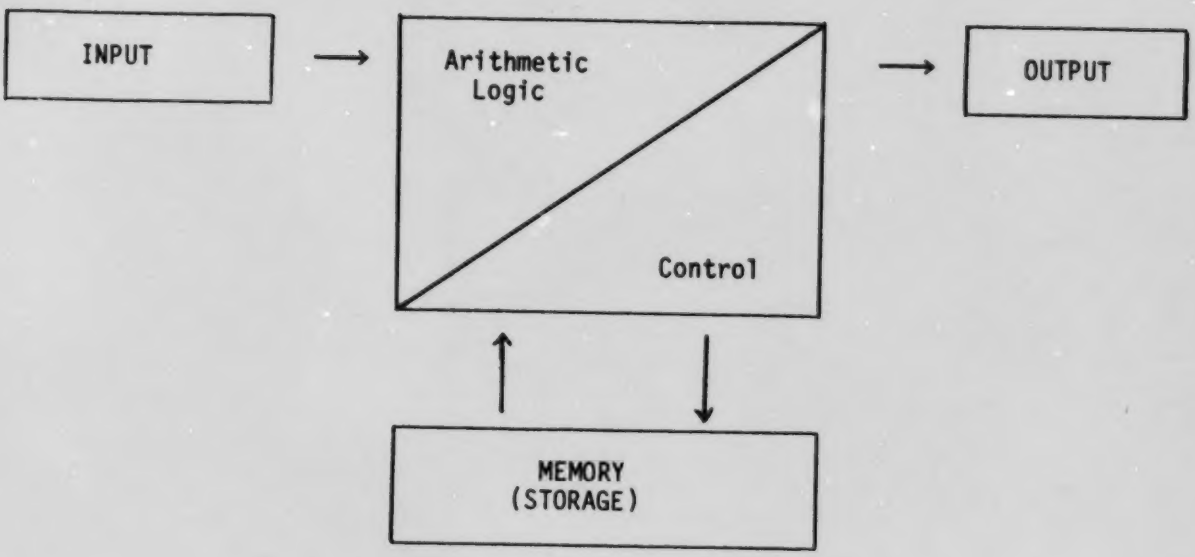
Supplement A3.2

Supplement A3.3

Supplement A3.4

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Films: Newton, David E. Slides on A Day In the Life of a Computer. Portland, Maine: J. Weston Walch, Publisher, 1974. INPUT/OUTPUT: How Computers Read and Write. Chicago, Illinois: Encyclopedia Britannica Educational Corporation, 1972.	

Block Diagram of a Computer System



Input/Output Media Activity

Classify the following items, placing the correct letter(s) after each word. Choose from one or more of the following: (A) input device (B) input medium (C) output device (D) output medium.

_____ 1. teletype

_____ 2. line printer

_____ 3. punched paper tapes

_____ 4. magnetic tape drives

_____ 5. card reader

_____ 6. paper tape reader

_____ 7. floppy disk

_____ 8. card punch

_____ 9. magnetic tape

_____ 10. console typewriter

_____ 11. punched cards

_____ 12. display terminal

ANSWER KEY TO SUPPLEMENT A3.2

1. A, C
2. C
3. B, D
4. A, C
5. A
6. A
7. B, D
8. C
9. B, D
10. A, C
11. B, D
12. C

Storing Information in a Computer

Some software (instructions stored in the memory of the computer) is developed by the computer manufacturer and others by the computer user. Let us see how information is stored in the computer. The particular form in which a computer processes information depends on the design of the computer and the coding scheme used. In general, however, computers process any piece of information as if it were a collection of 1's and 0's stored physically next to each other in what is called a computer "word." Using only 1's and 0's (binary digits) is the least expensive, most reliable device for storing information; one condition (like a switch in the ON position) represents a 1 and the other condition (like the OFF position of a switch) represents a 0. In today's computers these one-zero devices are electronic mechanisms called semiconductors.

Not all computer manufacturers use the same binary (one-zero) code. Cooperative efforts of computer users, computer manufacturers, and governmental agencies have resulted in the development of some standardized codes. The codes advocated by the U.S. government agencies is the American Standard Code for Information Interchange (ASCII). However, the major computer manufacturer, IBM, uses the code Extended Binary Coded Decimal Interchange Code (EBCDIC). Instructions and data can be converted from one code to another, but this requires time and additional programming.

Even if all characters were to have the same binary code in all computers, computer designers would still develop a set of instructions unique to each given computer. Individualized instructions enable users to take full advantage of the special features of a certain computer. Thus each computer has associated with it an instruction set (usually formulated in binary digits) called its machine language. To clarify the concept of machine language, one might compare it to the native language of a human being. Since machine language is always numeric and usually expressed in binary digits, languages have been devised to make it easier for humans to communicate with computers. Problem-oriented languages bridge the gap between man and machines. They use words and symbols specific to a given problem area.

One popular problem-oriented language is COBOL (Common Business Oriented Language). It is designed for use in a business and financial environment where large files of information are processed, usually to produce some kind of financial report. Consequently the language includes special words such as READ, MOVE, MULTIPLY, and WRITE that pertain to the handling of collections of information. Here is a portion of a COBOL program:

```
PROCEDURE DIVISION.
BEGIN.
      OPEN INPUT INPUT-DATA, OUTPUT OUTPUT-FILE.
      READ INPUT-DATA RECORD AT END GO TO JOB-DONE.
      WRITE OUT-LINE FROM HEADER AFTER ADVANCING TO TOP.
      MOVE ZERO TO MONTHS-FROM-NOW.
      MOVE NEXT-MONTH TO WHICH-MONTH.
      PERFORM CALCULATIONS ROUTINE.
      NUMBER-OF-MONTHS TIMES.
JOB-DONE.  CLOSE INPUT-DATA, OUTPUT-FILE, STOP RUN.
```

```
CALCULATION-ROUTINE.
      ADD 1 TO MONTH-FROM-NOW.
      IF WHICH-MONTH IS GREATER THAN 12 SUBTRACT 12,
      FROM WHICH-MONTH.
      MULTIPLY MONTHS-FROM-NOW BY PERCENTAGE
      GIVING SALES.
      ADD BASE TO SALES.
      MOVE MONTH(WHICH MONTH)TO MONTH NAME.
      WRITE OUT-LINE FROM PRINT-RECORD AFTER
      ADVANCING 1 LINES.
      ADD 1 TO WHICH-MONTH.
```

Even without knowledge of COBOL, you can see that this language uses words that make the program at least readable. Some computer users think COBOL requires too many words to define a given job.

Another popular problem-oriented language is FORTRAN (FORmula TRANslation). This language is used to solve problems in mathematics, chemistry, and physics, or any other problems that can be represented as formulas or equations. A FORTRAN program that computes the area of a triangle if the lengths of the base and height are given is the following:

```
READ,B,H
A=.5*B*H
PRINT,B,H,A
END
```

This program is more easily related to our BASIC II than the COBOL. In any problem-oriented language, however, the compiler (language translator) in the computer must convert the words and symbols into machine language.

ON-OFF=ONE-ZERO

If people knew how a computer's central processing device, its logic function, worked, they might be inclined to be less fearful and more friendly. Basically, a computer responds to the presence or absence of energy--a drumfire of impulses that turns it on and off. These are the only two states for a digital computer, which processes on-off as one-zero, yes-no and true-false. Thus, with its mentality restricted to on and off, a computer deserves more pity than scorn, for it is an extremely stupid machine. Computers can guide rockets to Mars, but they cannot multiply; they can calculate complicated bond yields, but they cannot subtract; they can prepare a payroll for a huge corporation, but they cannot divide. When it comes to numbers, all a modern digital computer can do is add. But it does this very fast, at the rate of millions of computations a second. What the "giant brains" lack in intelligence they make up for in speed.

The four arithmetic functions can all be expressed in terms of addition. Multiplication, of course, is merely continuous addition. Subtraction is the addition of the complement of the number to be subtracted. Division is the addition of complements. In the decimal system, the nines complement is used, which gets a bit cumbersome. But modern computers do not use decimals, they use the binary system: A number is either a one or a zero, and thus its complements are zero and one. Any number and hence any computation is processed as a series of ones and zeros.

In binary notation, a decimal 0 is also a binary 0, and a decimal 1 is a binary 1. But since a digital computer can handle only these two numbers, it has to express higher numbers as combinations of 0 and 1. Thus, 2 is expressed as a 1 and a 0 or 10; 3 is a 1 and a 1 or 11; 4 is a 1, a 0 and another 0 or 100. The three-digit decimal number 123 is, in binary code, 1111011. That is seven digits instead of three, but at computer speed the extra digits are hardly noticed; besides, computers have addition tables stored in their memories, which enables them to take short cuts.

From binary arithmetic to binary logic is a simple step. Essentially, the computer's two numbers now stand for "true" or "false." In an upgraded game of Twenty Questions--say, Twenty Million Questions--the computer can keep asking itself, "True or false?" In actual practice, the process is more sophisticated and allows for several alternatives. But still it is based on the binary inputs.

The combination of the simplest notation system, the binary, and high computation speeds enable the "stupid" computers to act like a very smart machine. Without the computer, we could never have put a man on the moon, for only computers could have calculated the necessary complex course corrections quickly enough. Without the computer, the telephone companies could not provide service at present rates, for they would have to add legions of employees to handle billing and switching functions that are now computerized. Without computers, a problem that a design engineer solves in seconds would take hours or days--and possibly would never be attempted at all.

Computers are a lot faster than they were, to be sure, but they are going to become still faster. According to Mark Shepherd, Jr., chairman and chief executive of Texas Instruments, computers are almost at the midway point of their theoretical limit in improvement with respect to speed and cost. During the past thirty

years, Shepherd points out, computer computation has gone from a few instructions per second at a cost of millionths of a cent each. The same magnitude of change, he says, lies ahead.

"On-Off=One-Zero," Flighttime, Allegheny Airlines, June 1978.

Objective A4: List and describe computer-related careers, stating educational and experience requirements, job opportunities, and current salary ranges.

Textbook Reference: Pp. 128-129 - The Computer Operator
Pp. 240-241 - The Computer Programer
Pp 396-397 - The Systems Analyst

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Students need to take a look at the entire spectrum of computer-related jobs. Usually this is best done on an individual basis, but reports by various members of the class and/or speakers can be effective in achieving this objective. Encourage students to use the classified section of the newspaper and publications such as Computer World to locate career information. The jobs listed in Supplement 4.1 may be altered according to trends and demands in the industry. Often students find part time or summer jobs involving computers through this project. This can be a long-term assignment, thus encouraging students to budget their time and to make use of class time when there is no available computer. Students may be encouraged to present their reports in original forms, such as charts, graphs, mobiles, or posters.	Supplement A4.1

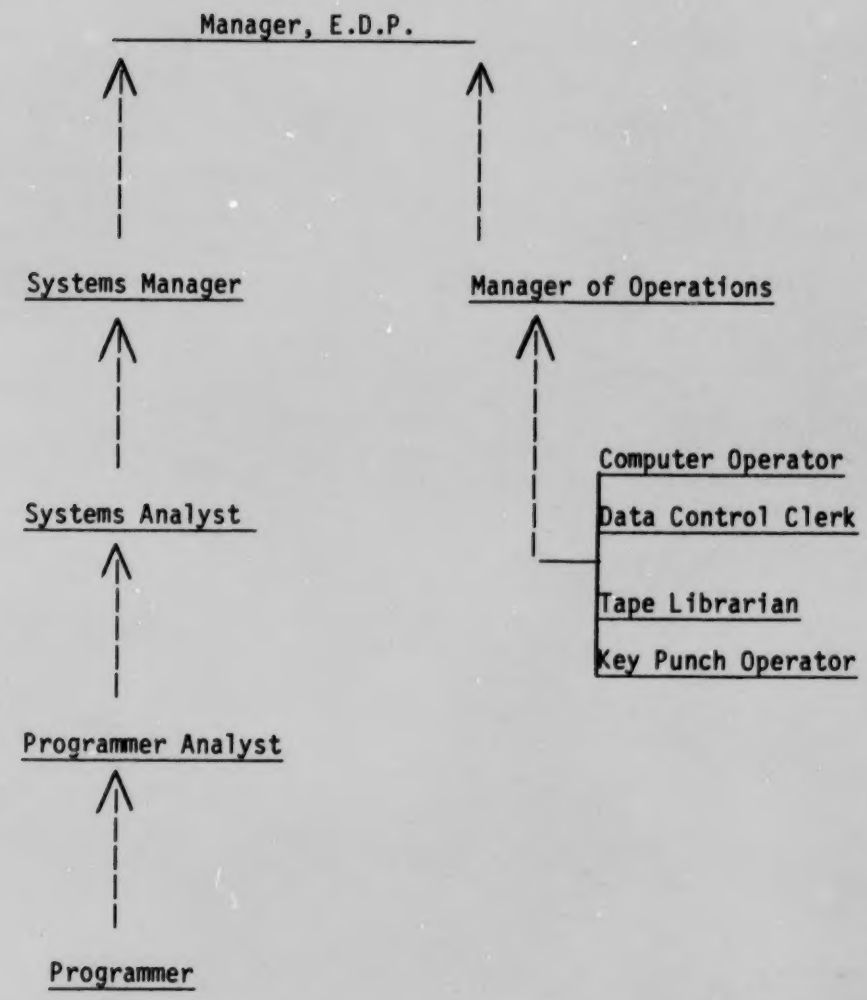
Occupations Project

Listed below are ten computer-related jobs. Find the following information about each of these jobs: a) salary range, b) educational and experience requirements, and c) duties and responsibilities.

Key punch operator	Programmer analyst
System analyst	Systems manager
Manager of operations	Data control clerk
Tape librarian	Computer operator
Programmer	Manager, e.d.p.

Create an organization chart, showing possible paths of advancement. (Some jobs may be on the same level.)

SAMPLE ORGANIZATION CHART



Objective A5: The student will write a paper on a computer-related topic.

Textbook Reference:
P. 78 – Computers and the Visually Handicapped
Pp. 180-181 – Speech Recognition by Computer
P. 191 – Image Processing by Computer
Pp. 234-235 – Thesaurus Linguae Graecae
P. 299 – Computer-Assisted Legal Researcher
Pp. 320-321 – Artificial Intelligence
Pp. 368-369 – Computerized Fingerprint Analysis

TEACHING SUGGESTIONS

Encourage students to keep abreast of computer developments. A weekly trade journal such as Computer World or a computer magazine such as Infosystems can be helpful, but they are often too technical; there is adequate coverage in newspapers and on TV. Fill the bulletin board with articles about data processing, or let the students do it themselves (even for extra credit, if desired).

Supplement 5.1 is a sample assignment. Students should discover that computers will be involved in any career they might choose, from housewife to auto mechanic to brain surgeon. There are other options for students undecided on a career. As with the previous objective, this can be a long-term assignment, one to work on when there is no computer time.

STUDENT ACTIVITIES IN THIS GUIDE

Other textbook references:

P. 377 – The National Crime Analysis Information Center

P. 411 – Computerized Cash Control at Burger King

Pp. 438-439 – Computers in Aviation

Supplement A5.1

You and Computers

Perhaps you have already chosen a college major or a career. If so, research the use of data processing in your chosen field. Otherwise, choose a computer application of interest to you or perhaps an area which involves someone in your family. Almost any field of interest will have some involvement with data processing. Include paragraphs on the following topics:

(A) Present state of data processing in your field

(B) Three process or equipment innovations in the area

(C) The future of data processing in this field

A bibliography should be included.

Objective A7: The student will use the system commands necessary to enter and run a program.

Textbook Reference: Topic 1, pp. 1-35

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Every student should be able to operate the computer system before any programming groups are formed. (See Preface.) They should be able to initialize the system and name and identify all its parts, such as the CPU and CRT. To ensure that each student has reached this level, an operator test may be given. Select from a number of simple programs. (See Supplement 7.1.) To test each student individually is time-consuming, but this procedure can eliminate later problems and alert the teacher to students who have not comprehended the initial lessons. To produce the desired objective, the students may be asked to repeat the operator test until they perform satisfactorily, demonstrating a basic knowledge of the hardware and the software. This operator test not only gives one insight into the individual student's progress, but it also gives confidence to the students once it has been successfully completed. It is unlikely that the available system operates exactly as described in the textbook. Check system manual for exact format needed.	Supplement A7.1 - Hands-on Operator's Test. The teacher should supply the test data. Supplement A7.2 - (Region IV) - LOG-ON and LOG-OFF Procedures - Have each student log-on and log-off when taking operator's test. Afterwards, however, only the first student on the computer each day should LOG-ON and only the last student should LOG-OFF. Supplement A7.3 - System Commands Used with Region IV. Supplement A7.4 - Test for Topic I (Sample Test)

Operator's Test

Sample programs to use - either supply the data or ask the student to do so.

*Write a program to

1. Average 5 grades.
2. Compute the perimeter of a rectangle, given the length and the width.
3. Compute the area of a triangle, given its base and height.
4. Calculate the mileage of a car traveling ___?___ miles using ___?___ gallons of gasoline.
5. Compute the area of a circle, given the radius.
6. Find the 3rd angle of a triangle, if given two of the angles.
7. Calculate the interest (simple) on a loan, given the principal and the rate.
8. Find the average cost of four records, given each individual price.
9. Given two points, find the slope of the line connecting them.
10. Use the Pythagorean theorem to solve for the hypotenuse if given the legs of a right triangle.

*Choose programs according to the student's math background.

Region IV LOG-ON and LOG-OFF PROCEDURES

To LOG-ON

1. Dial phone number assigned to computer.

2. Set coupler switches to "ON," "FULL," "ORIGINATE," "ACOUSTICAL."

3. When computer answers phone with shrill whistle, place phone in coupler. If coupler is separate unit, cords of coupler and phone should be at same end. If coupler is built into terminal, phone cord should be toward rear of coupler.

4. Turn terminal switch to "LINE." Red and green lights must be on.

5. Hit the carriage return key (CR) .

6. System will print a short message followed by "USER NUMBER." Respond by typing the school's assigned used number and hitting (CR) .

7. System will print "PASSWORD." Respond by typing assigned password and hitting (CR) .

8. System will then print "RECOVER/CHARGE."

   a) If a new LOG-ON procedure is used, begin work by typing "CHARGE" followed by a comma and the assigned charge number. Then hit (CR).

   b) If connection was inadvertently lost and the used is attempting to reconnect and recover work in progress, type "RECOVER" followed by a comma and the assigned charge number. Then hit (CR) .

9. The system prints a short message followed by "READY." Respond by typing "BASIC" or other language to be used and hitting (CR).

10. System prints "OLD, NEW, OR LIB FILE." Respond with appropriate type of program:

    a) "OLD" if program was previously saved by user.

    b) "NEW" if a new program is to be written.

    c) "LIB FILE" if a program is to be called up from the system library.

    Hit (CR).

11. System prints "FILE NAME."

    Respond by giving name of program. Name may have no more than seven alphanumeric characters and must begin with a letter. Hit (CR) .

    If an "OLD" or "LIB FILE" program has been indicated, the system searches the files for this file name. If the file does not exist, the system responds "FILE NOT FOUND."

2. After proper procedure is used in naming of program file, system responds with "READY."

   a) If a "NEW" program was indicated, the computer is ready for the program to be entered.

   b) If the program is on paper tape, follow procedure given Topic 2, Supplement 12.3

   c) If an "OLD" program was indicated, computer has located program and needs directions to proceed.

13. Continue with work to be done. While still on line, begin another program by simply hitting (CR) and start at step 9. Each time a different student begins working, it is NOT necessary to LOG-ON again.

    The first user should LOG-ON.

    The last user should LOG-OFF.

To LOG-OFF

1. Type "BYE" and his (CR).
2. System will print LOG-OFF messages.
3. Wait for green light to go off.
4. Lift phone from coupler.
5. Turn both terminal switch and coupler switch to OFF position.

COMMANDS USED WITH REGION IV

1. AUTO - Automatically numbers the statements.
   Use the ESC key to end the process.
2. LIST - Makes a list of a program exactly as the computer has it stored.
3. LNH - Same as LIST except it omits the heading.
4. NEW - Prepares computer for input of a new program.
5. OLD - Permits use of a program stored in the user's file.
6. PURGE - Used to erase a stored program.
7. RECOVER - A LOG-IN command used to recover after an accidental LOG-OFF.
8. RESEQUENCE - Reassigns line numbers to the primary file, starting at 100 and advancing in intervals of ten.
9. RNH - Same as RUN but omits the heading.
10. RUN - Has the computer execute the program.
11. SAVE - Enters program into the user's catalog.
12. SCRATCH - Destroys the program but leaves the names intact. It gives the user a clean sheet on which to work.
13. STOP - Stops computation at once.
14. TAPE - Prepare computer to read a file in tape form.
15. STATUS - Gives an indication of the status of the operation of computer (running, idle).

Specific commands used in error correction:

Printing loop - Depress BREAK key.

Then type STOP.

Computing loop - Type STOP.

To delete errors:

a) Line delete - Depress ESC key (before sending line).

b) Line delete - Retype line number (after sending line).

c) Character delete - Depress ← for each character or space to be deleted.

Test 2 for Topic I

I. List 3 types of input media that could be used in a computer system.

1. _____

2. _____

3. _____

II. List 3 types of output devices that could be used in a computer system.

1. _____

2. _____

3. _____

III. Use a diagram to describe the components of a computer system.

IV. Choose the correct letter.

_____ 1. The slide rule was developed as an outgrowth of
A) the abacus B) logarithms C) the soroban D) none of these

_____ 2. The punched card principle for weaving patterns in rugs on the first automatic loom was perfected by
A) Bouchon and Falcon B) Herman Hollerith C) Joseph Jacquard
D) James Powers

_____ 3. In 1880, the U.S. Census Bureau employed this person to speed up the sorting and tabulating of census data:
A) James Powers B) Howard Aiken C) Charles Zavier Thomas
D) Herman Hollerith

_____ 4. The first man to describe a machine similar to the modern computer was
A) Powers B) von Leibnitz C) Hollerith D) Babbage

_____ 5. The first truly electronic digital computer was developed in
A) 1890 B) 1946 C) 1931 D) 1954

_____ 6. The elctronic computer in which the stored-program concept was perfected was
A) MARK I B) ENIAC C) Babbage D) EDVAC

_____ 7. The first adding machine was developed by
A) Babbage B) Napier C) Pascal D) Hollerith

_____ 8. A computer which performs operations by actual counting is known as the
A) digital computer B) analog computer C) magnetic bubble computer D) fluidics computer

_____ 9. The event which hurried the development of the first generation of electronic computers was
A) World War I B) World War II C) the Space Age
D) The Cold War

_____ 10. The forerunner of the digital computer was
A) the slide rule B) the abacus C) the calculator D) ENIAC

_____ 11. The machine language of a computer is usually formulated in
A) base 10 B) COBOL C) binary digits D) a human language

_____ 12. The first generation of electronic computers used
A) transistors B) gears and cogwheels C) vacuum tubes
D) microcircuits

_____ 13. The most widely used programming language is
A) BASIC B) COBOL C) FORTRAN D) ALGOL

_____ 14. The first all electronic computer was the
A) MARK I B) UNIVAC C) ENIAC D) IBM 360

_____ 15. The first computer to sell commercially was the
A) MARK I B) UNIVAC C) TRS-80 D) IBM 360

_____ 16. The physical equipment comprising a computer is known as the
A) software B) program C) hardware D) component

_____ 17. A source document is one
A) on which output is recorded B) on which input is recorded
C) from which raw data is obtained D) none of these

TRUE/FALSE

_____ 18. The computer has made the Space Age possible.

_____ 19. The computer is capable of making accurate decisions without man's assistance.

_____ 20. Computers are capable of correcting errors in the data and instructions which they receive.

_____ 21. Data processing is the manipulation of factual matter of all kinds in order to produce a desired answer or result.

_____ 22. The chief goal of data processing is the conversion of raw data to useful information.

_____ 23. Automated data processing can only be understood if one has a knowledge of the internal design of the machines used.

_____ 24. The B in BASIC stands for basic.

_____ 25. Output is the processed information that flows from a data processing system.

_____ 26. A READ statement can appear anywhere in the program before the END statement.

_____ 27. There can not be a valid READ statement in a program without a DATA statement.

_____ 28. PRINT "X" will print the numeric value of the variable x.

_____ 29. RUN and LIST are commands, and never appear as statements in a program.

_____ 30. Typing in NEW and hitting the CLEAR KEY produce the same results.

_____ 31. All calculations must be made to the left of the = sign in a BASIC assignment statement.

_____ 32. The LET in an assignment statement is not necessary.

_____ 33. Typing O and Ø mean two different things to our machines.

_____ 34. A DATA statement must always end in a comma.

_____ 35. The END statement is the last statement in a program.

ANSWER KEY
Test 2 for Topic I

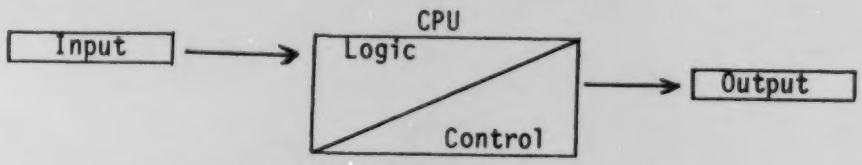
I. List 3 types of input media that could be used in a computer system.

1. punched cards
2. magnetic tape
3. punched paper tape

II. List 3 types of output devices that could be used in a computer system.

1. line printer
2. card punch
3. magnetic tape drive/paper tape punch

III. Use a diagram to describe the components of a computer system.



IV. Choose the correct letter.

B 1. The slide rule was developed as an outgrowth of
A) the abacus B) logarithms C) the soroban D) none of these

C 2. The punched card principle for weaving patterns in rugs on the first automatic loom was perfected by
A) Bouchon and Falcon B) Herman Hollerith C) Joseph Jacquard
D) James Powers

D 3. In 1880, the U.S. Census Bureau employed this person to speed up the sorting and tabulating of census data:
A) James Powers B) Howard Aiken C) Charles Zavier Thomas
D) Herman Hollerith

D 4. The first man to describe a machine similar to the modern computer was
A) Powers B) von Leibnitz C) Hollerith D) Babbage

B 5. The first truly electronic digital computer was developed in
A) 1890 B) 1946 C) 1931 D) 1954

D 6. The elctronic computer in which the stored-program concept was perfected was
A) MARK I B) ENIAC C) Babbage D) EDVAC

C 7. The first adding machine was developed by
A) Babbage B) Napier C) Pascal D) Hollerith

A 8. A computer which performs operations by actual counting is known as the
A) digital computer B) analog computer C) magnetic bubble computer D) fluidics computer

B 9. The event which hurried the development of the first generation of electronic computers was
A) World War I B) World War II C) the Space Age
D) The Cold War

B 10. The forerunner of the digital computer was
A) the slide rule B) the abacus C) the calculator D) ENIAC

C 11. The machine language of a computer is usually formulated in
A) base 10 B) COBOL C) binary digits D) a human language

C 12. The first generation of electronic computers used
A) transistors B) gears and cogwheels C) vacuum tubes
D) microcircuits

B 13. The most widely used programming language is
A) BASIC B) COBOL C) FORTRAN D) ALGOL

C 14. The first all electronic computer was the
A) MARK I B) UNIVAC C) ENIAC D) IBM 360

B 15. The first computer to sell commercially was the
A) MARK I B) UNIVAC C) TRS-80 D) IBM 360

C 16. The physical equipment comprising a computer is known as the
A) software B) program C) hardware D) component

C 17. A source document is one
A) on which output is recorded B) on which input is recorded
C) from which raw data is obtained D) none of these

TRUE/FALSE

T 18. The computer has made the Space Age possible.

F 19. The computer is capable of making accurate decisions without man's assistance.

F 20. Computers are capable of correcting errors in the data and instructions which they receive.

T 21. Data processing is the manipulation of factual matter of all kinds in order to produce a desired answer or result.

T 22. The chief goal of data processing is the conversion of raw data to useful information.

F 23. Automated data processing can only be understood if one has a knowledge of the internal design of the machines used.

F 24. The B in BASIC stands for basic.

T 25. Output is the processed information that flows from a data processing system.

T 26. A READ statement can appear anywhere in the program before the END statement.

T 27. There can not be a valid READ statement in a program without a DATA statement.

F 28. PRINT "X" will print the numeric value of the variable x.

T 29. RUN and LIST are commands, and never appear as statements in a program.

F 30. Typing in NEW and hitting the CLEAR KEY produce the same results.

F 31. All calculations must be made to the left of the = sign in a BASIC assignment statement.

T 32. The LET in an assignment statement is not necessary.

T 33. Typing O and Ø mean two different things to our machines.

F 34. A DATA statement must always end in a comma.

T 35. The END statement is the last statement in a program.

Objective A10: The student will debug a BASIC program of 10 or more statements.

Textbook Reference: Pp. 12-28

TEACHING SUGGESTIONS

Stress that computers cannot think and cannot read minds. The computer can only "read" exactly what is written; therefore, it is absolutely necessary to use care in typing.

Notes

1. This textbook assumes that the word LET must be included in assignment statements. Check system to see whether this is really necessary.
2. It is a good practice to use END as the last statement of all programs even though some systems do not require it.
3. Use of multiples of 10 for line numbers makes inserting lines simpler.
4. Editing features (if available) can be substituted. If students can delete, insert, and change characters in the edit mode, they will spend less time entering their programs.

Teletypewriter references may be ignored if only microcomputers are available.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral

Exercise Two, p. 13, #1-2
Exercise Three, pp. 22-23, #1-4
Objective Activity, pp. 26-28, #3-14

Written

Problem Set One, pp. 32-35, #1-13

Program

Problem Set One, p. 35, #14, 15, 17

An alternate procedure is:

Students can be assigned all of the Problem Set at the back of the chapter (these are the only problems where the answers are not given). Use the exercises and objective activities as oral exercises. The programming part of the Problem Set (the last problems) can be assigned as a group activity.

TOPIC TWO: ELEMENTS OF BASIC

Performance Objectives:

The student will -

A9 - Make traces of programs with ten or more statements in order to determine their output.

A11 - Compute values of arithmetic statements involving addition, subtraction, multiplication, division, exponentiation, and parentheses according to predefined hierarchy of operations.

A12 - Write programs which utilize LET, PRINT, END, REM.

References:

Textbook pages 37-66

Objective A9: The student will make traces of programs with 10 or more statements.

Textbook Reference: Pp. 37-45, 64

TEACHING SUGGESTIONS

Spend a short time going over necessity of exactness in writing BASIC statements. Remind students that the computer is "dumb." It does not understand what they are thinking.

Concentrate on tracing short programs, involving only LET, PRINT, END. A good understanding of tracing (called "determining the output" in the textbook) is necessary for debugging of more complicated programs.

Check to see what form BASIC variables may take on the system being used. The legal BASIC variables, as defined on p. 40, are very limiting (single letters and/or single letters followed by 1 digit). Most systems allow for 2 letter variables, (i.e., AB), as well. An expansion of the legal BASIC variables will cause many answers to differ from the textbook in subsequent exercises.

STUDENT ACTIVITIES IN THIS GUIDE

Oral

Exercise One, p. 41, #2, 3

Exercise Two, p. 45, #1, 2

Written

Problem Set Two, p. 64, #1, 2

Objective A11: The student will compute the value of arithmetic statements involving addition, subtraction, multiplication, division, exponentiation, and parentheses according to the predefined hierarchy of operations.

Textbook Reference: Pp. 42-51, 64

TEACHING SUGGESTIONS

Review algebraic operations. Be sure students have a grasp of the hierarchy of operations.

(Helpful: Please Excuse My Dear Aunt Sally.)

Emphasize that ALL operators must be printed. NONE can be assumed.

Be sure that students are introduced to ↑ and ^ as methods of exponentation.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral

Exercise Two, p. 45, #3, 4
Exercise Three, p. 51, #1-2

Written

Problem Set Two, p. 64, #3, 4

Note: The problems throughout the textbook are not oriented toward students of higher mathematics. Alternate problems may be found in other texts such as Neal Golden's Programming in the BASIC Language and James S. Coon's Basic Basic (see Bibliography).

Guide

Supplement A11.1
Supplement A11.2
Supplement A11.3

ORDER OF OPERATIONS - WORKSHEET 1

Following the rank of operations on p. 40, perform the following calculations at your desk. Check your answers on the computer. Remember that operations in parentheses must be performed first, regardless of the hierarchy of operations.

1. PRINT 6 + 3 * 2
2. PRINT 2/5 * 2
3. PRINT 6 + 8 - 2
4. PRINT 2 + 3 - 2
5. PRINT 5 * 2 - 3 * 3
6. PRINT 6 * 2 + 3
7. PRINT (3 + 2) * 5
8. PRINT 24 / (4 + 2)
9. PRINT (5 + 2) + 2
10. PRINT (3 + 1) + (2 + 1)
11. PRINT 6 + 8/2 - (2 * 4)
12. PRINT 11 - 2 * (3 + 1)
13. PRINT (8 - 5) * 3
14. PRINT 15/ (3 * 5)
15. PRINT (6 - 2 + 1) * 3 + 2

ANSWER KEY

SUPPLEMENT A11.1

1. 12
2. .8
3. 12
4. 6
5. 1
6. 48
7. 25
8. 4
9. 49
10. 64
11. 2
12. 3
13. 9
14. 1
15. 45

ORDER OF OPERATIONS

WORKSHEET 2

I. Write a BASIC statement for each of the following mathematical expressions.

a. 23×32

b.
$$\begin{array}{r} 2419 \\ 3347 \\ 1989 \\ +\ 4434 \\ \hline \end{array}$$

c. $2^3 + 3^2$

d. $2 + 4(3-2)$

e.
$$\begin{array}{r} 0.3275 \\ \times\ .521 \\ \hline \end{array}$$

f. $\dfrac{4 + 6}{3 + 2}$

II. Change the following BASIC expressions to mathematical expressions and evaluate.

a. 3 ∧ 4 + 6

b. 5 * (2 + 8)

c. 8 * 3 ∧ 2

d. 3 * 4

e. 3 + 6 * -4

f. 2 * 2 * 2 + 4

g. 4 * 5 + 2 ∧ 3

h. 6 + 4/2

i. 5 ∧ 2 + 6

j. 4 + 6 ∧ 2

III. Write the answers to the following BASIC expressions.

a. 50/5 ∧ 2

b. 4 + 2 * 6

c. (5 - 2) * 7 ∧ 2

d. 3 * (6-3) * 45

e. 4 ∧ 2 * 6 - 2

f. 25 * 4 - 8/2 ∧ 3

ANSWER KEY TO SUPPLEMENT A11.2

I. a. 23 * 32

b. 2419 + 3347 + 1989 + 4434

c. 2 ∧ 3 + 3 ∧ 2

d. 2 + 4 * (3-2)

e. 0.3275 * .521

f. (4 + 6) / (3 + 2)

II. a. $3^4 + 6 = 87$

b. $5 (2 + 8) = 50$

c. $8 \times 3^2 = 72$

d. $3 \times 4 = 12$

e. $3 + 6 \times 4 = 27$

f. $2 \times 2 \times 2 + 4 = 12$

g. $4 \times 5 + 2^3 = 28$

h. $6 + 4 \div 2 = 8$

i. $5^2 + 6 = 31$

j. $4 + 6^2 = 40$

III. a. 2

b. 16

c. 147

d. 405

e. 94

f. 99

Hierarchy Problems

Find the value of each variable:

1. 10 LET A = 5 * 2 + (3/6 * 2 + 5 - 12) * 3/2

2. 20 LET B = 4 + (2 *(3/1))/6 + 12

3. 30 LET C = 4 * 6 - 4/2/2 * 6/2 - 1

4. 40 LET D = ((2 2/2 - 1) * 7) + 6 - 5 * 2

5. 50 LET E = 5 - 2 * (3*(4+6)/5 - 2 3 + 9/(4-1) + 6 - 4 (5 - 3)

ANSWER KEY TO SUPPLEMENT A11.3

1. A = 1

2. B = 17

3. C = 20

4. D = 3

5. E = 23

Objective A12: The student will write and flowchart BASIC programs which utilize LET, PRINT, END, REM.

Textbook Reference: Pp. 51-66

TEACHING SUGGESTIONS

First have students write BASIC assignment statements involving hierarchy of operations.

After students are able to write and correct individual statements, put these statements into short programs.

Explain and give examples of:

REM
LET
PRINT
END

Review with students the system commands necessary to put programs into the system(s) used.

If there is still no computer accessibility, supplement 12.1 may be useful.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral and Board Work

Exercise Three, p. 51, #3

Exercise Four, p. 54, #1, 2, 3, 4

Written

Problem Set Two, pp. 65-66, #5a-i

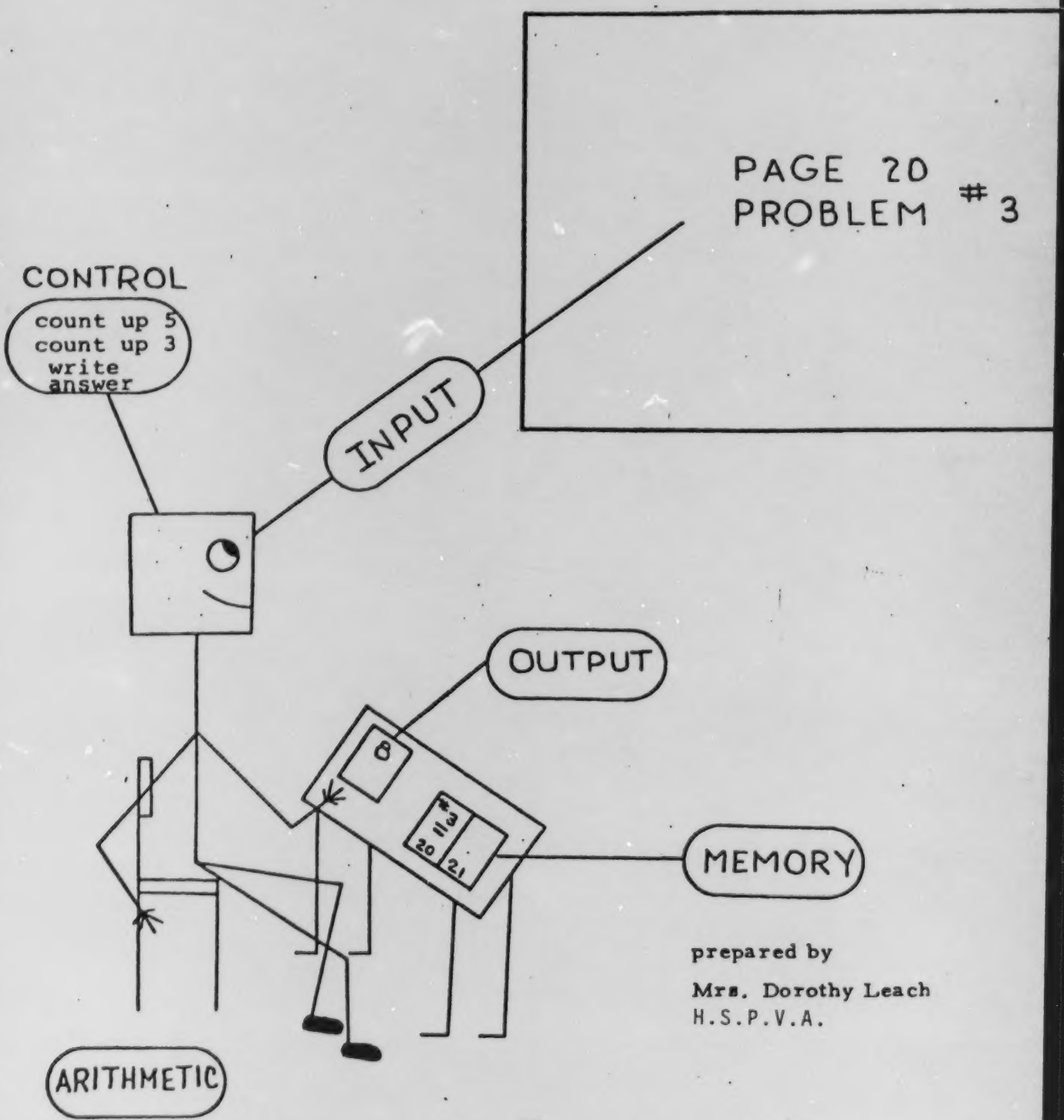
Programs

Problem Set Two
Page 66, #6

Note

If Region IV is being used but there is still no hook-up, programs can be written on paper tape for later processing.

(Supplement 12.2)
(Supplement 12.3)



prepared by

Mrs. Dorothy Leach
H.S.P.V.A.

Remember - far back in your past....back to the first grade. The teacher has just made an assignment. "Add the following two numbers."

```
  5
 +3
```

The diligent students get out pencil and paper and start to work. The not-so-diligent student in the back of the room (could it be you?) has not memorized his addition facts. He tries another approach - he sticks his hands under the desk, counts out 5 on one hand, 3 on the other, counts up the answer, and quickly writes it down before the teacher catches on that he hasn't learned to add!

The computer you are learning to program can be compared to the not-so-diligent student. The first grader can read (a little); so can the computer, but it's vocabulary is limited, and it has no imagination at all. Since it can read, it can follow whatever instructions we give it, just as the first grader did when the teacher wrote the assignment on the board. The instructions to the computer must be given in its very limited language. It will carry out these instructions exactly, even if the instructions contain a very obvious mistake. This "reading" function is called input.

The first grader can do arithmetic by counting on his fingers. The computer can also perform arithmetic, and in the same way as the first grader. Fortunately, the computer has many more places to count numbers than the student has fingers, and fortunately the computer can count up the sum much faster than the student can, but it's method of solving the problem is the same as the first grader's.

The first grader decided what to do, and when to do it after he read his instructions. Part of the computer performs a similar control function, following instructions at the correct time, translating commands into action.

If the teacher had made the assignment this way:

"turn to Page 20 and do Problem #3"

the first grader would have looked up the problem in his book and then followed the instructions. The textbooks and all other ways of recording information form a kind of memory for human beings. Computers also contain a memory, and can store and recall large amounts of information in this memory.

When the first grader had finished working the problem, he wrote down the answer where it could be read by his teacher. The computer must also be able to communicate results where they can be seen. This function is called output.

The computer then is like a not-so-diligent but infinitely patient first grader. It reads a little, and always does exactly what it is told. It never gets tired or bored, even when it does the same thing over and over again. It has a large memory, and can recall instantly the facts it has stored in its memory. Best of all, it can do calculations with incredible speed, without mistakes, all by counting.

Go back to the first grade, back to the student in the corner of the room. How does he solve this problem? Could you have made a set of instructions (an algorithm) to show another first grader how to solve a problem this way?

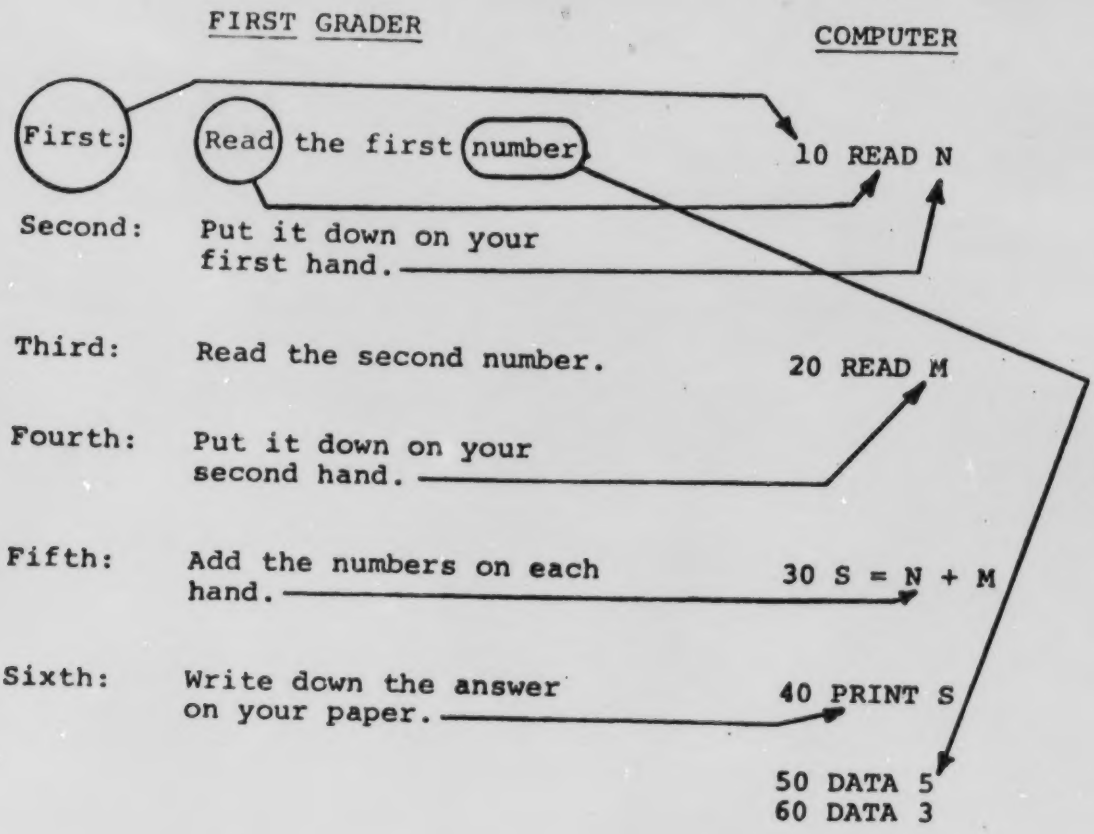
HOW TO ADD TWO NUMBERS

First:	Read the first number.
Second:	Put it down on your first hand.
Third:	Read the second number.
Fourth:	Put it down on your second hand.
Fifth:	Add (count up) the numbers on each hand.
Sixth:	Write down the answer on your paper.

A set of instructions in a computer language is called a program. A program might look like this:

```
10 READ N
20 READ M
30 S = N + M
40 PRINT S
50 DATA 5
60 DATA 3
```

Compare the two sets of instructions.



The program uses four BASIC instructions: READ, DATA, =, and PRINT.

```
10 READ N
20 READ M
 .
 .
50 DATA 5
60 DATA 3
```

Tells the computer to look for a DATA statement in the program

where the number to be used in the problem will be found.

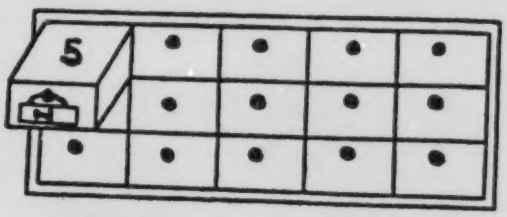
10 READ N This means READ a number from a DATA statement and store it someplace. Call the place N.

It is easier to picture what is happening if you imagine part of the computer as a large cabinet full of drawers. Each drawer can hold a number, and each drawer can have a name hung on it. The names of the locations (drawers) where numbers are stored are called variables.

The instructions

```
10 READ N
 .
 .
 .
50 DATA 5
```

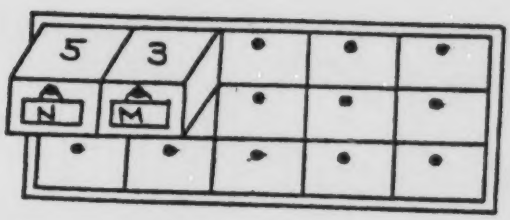
cause the computer to find a "drawer" (a location), store the number 5 in it, and hang the name N on the "drawer."



The instructions

```
20 READ M
 .
 .
 .
60 DATA 3
```

cause the computer to store 3 in a location called M.



A READ statement may contain more than one variable.

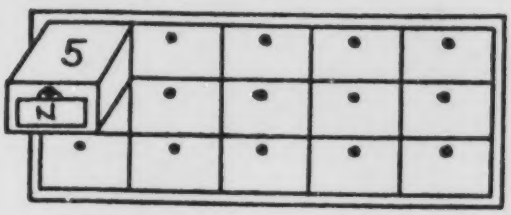
```
10 READ N
20 READ M      or      10 READ N,M
```

The variables are separated by commas. A DATA statement may contain more than one number. The numbers are also separated by commas.

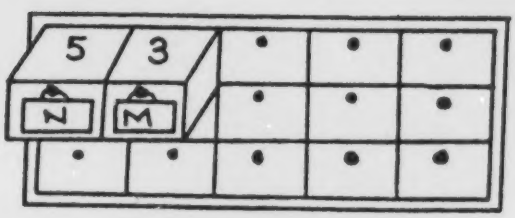
```
50 DATA 5
60 DATA 3      or      50 DATA 5,3
```

When READ or DATA statements contain more than one item, the first number in the first DATA statement is stored in the first variable in the first READ statement.

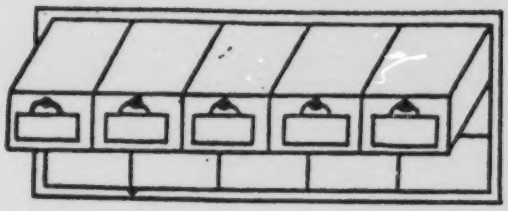
```
10 READ N,M
    .
    .
    .
50 DATA 5,3
```



The second number is stored in the second variable:



and so on:



The program to add two numbers could look like this:

```
10 READ N,M
20 S = N + M
30 PRINT S
40 DATA 5,3
```

Write a program in BASIC to subtract one number from another.

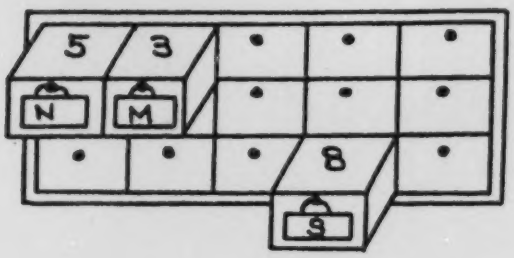
A program to do this might look like:

```
10 READ N,M
20 S = N - M
30 PRINT S
40 DATA 5,3
```

The second instruction in the adding program

```
20 S = N + M
```

is an arithmetic statement. It causes the computer to add the numbers in N and M and put the result in a location called S.



The symbol = in BASIC means put the number named on the right

```
S = [N + M]
```

in the place named on the left:

```
[S] = N + M
```

Arithmetic statements may contain either numbers or variables on the right side of =. A statement may contain only variables, as in the adding program.

```
20 S = N + M
```

or only numbers

```
390 T = 5 + 3.9
```

or both

```
75 R = 2 - X
```

Only a single variable may be on the left of =.

The operators in an arithmetic statement are:

+	which means plus
-	minus
*	times
/	divided by
↑or^	raised to the power
( )	inclusion

Write a program to divide a number by another.

A BASIC program to divide a number by another might look like this:

```
10 READ N,M
20 Q = N/M
30 PRINT Q
40 DATA 12,3
```

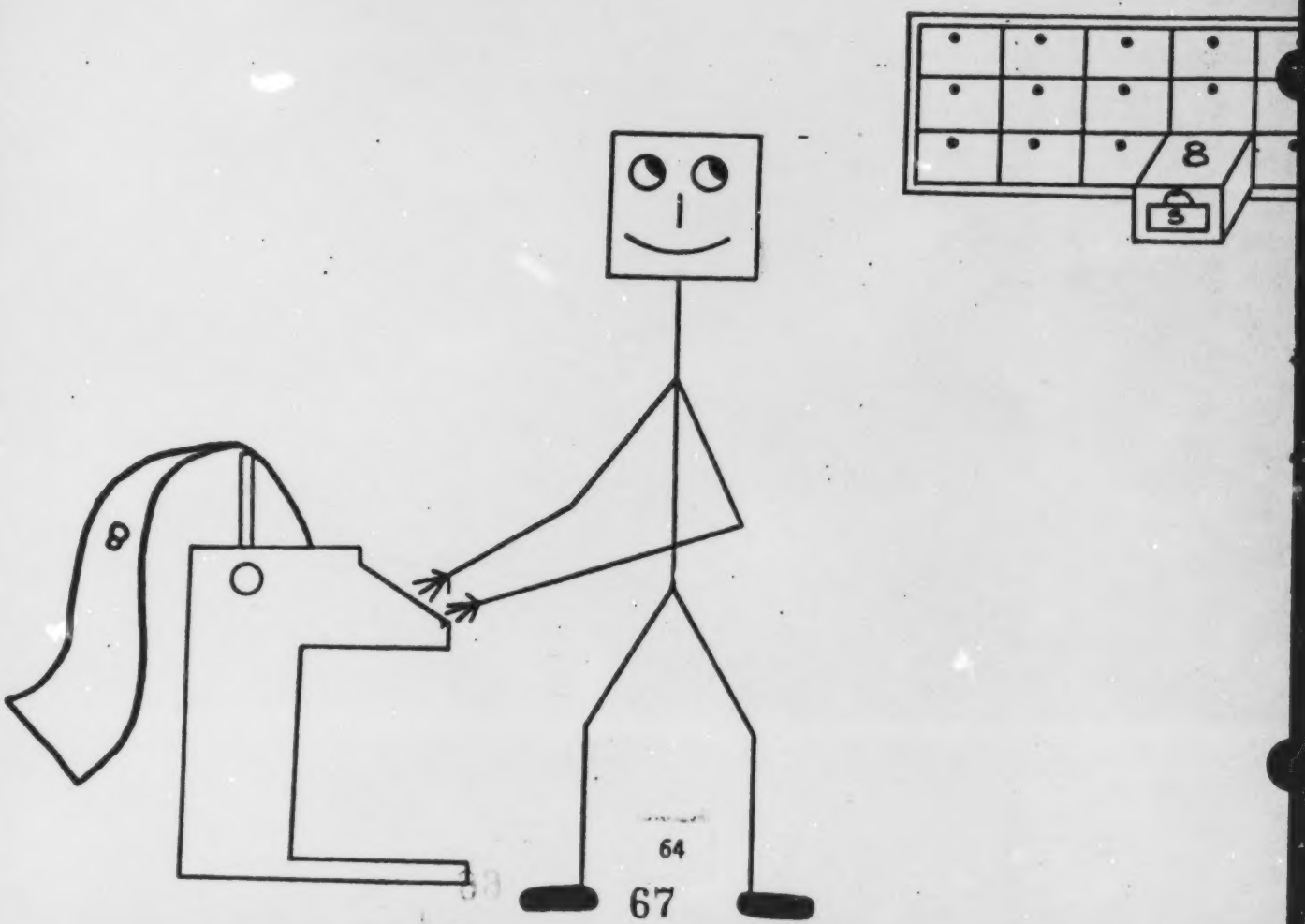
The following program finds the quotient and the product of two numbers:

```
10 READ W,Y
20 Q = W/Y
30 P = W*Y
40 PRINT Q
50 PRINT P
60 DATA 20,4
```

The instruction

```
30 PRINT S
```

causes the computer to "look at" the number in S and type that number on the teletype.



A PRINT instruction may ask that the contents of more than one variable be typed.

```
30 PRINT A,B,S
```

As usual, the variables are separated by commas. The teletype would type:

```
 5     3     12
```

separated by spaces.

Write a program to add, subtract, multiply, and divide two numbers and type all the answers.

This is one way to write the program:

```
10 READ A,B
20 S = A + B
30 D = A - B
40 P = A * B
50 Q = A / B
60 PRINT S,D,P,Q
70 DATA 30,6
```

Each instruction in a BASIC program starts with a number, called a line number.

```
10 READ N,M
20 S = N + M
30 PRINT S
40 DATA 5,3
```

Line numbers tell the order in which the instructions are to be followed.

```
40 DATA 5,3
20 S = N + M
30 PRINT S
10 READ N,M
```

would be done in exactly the same order as:

```
10 READ N,M
20 S = N + M
30 PRINT S
40 DATA 5,3
```

Four instructions; READ, DATA, =, and PRINT are enough to write a program which, when input to a computer, will cause it to add two numbers.

But WHY! Who would write a program, which takes a long time, to do a problem a six year old could do in a minute?

Nobody would. But somebody would write this program:

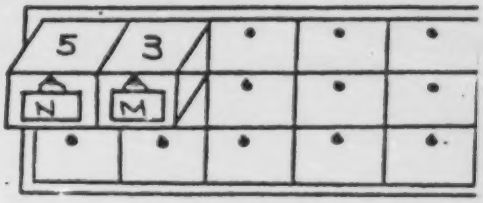
```
10 READ N,M
20 S = N + M
30 PRINT S
40 GØ TØ 10
50 DATA 5,3,6,9,4,7.1,-5,....
60 DATA 3.9,6.8,-1,-9.6,....
.
.
9610 DATA ..............2,4
```

This program tells the computer to add many pairs of numbers, to keep adding pairs of numbers until it runs out of data. The trick is instruction 40.

```
40 GØ TØ 10
```

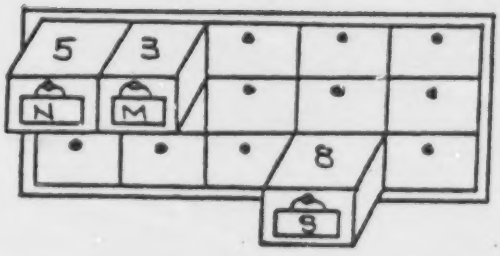
Instructions in a program are carried out in order of LINE NUMBERS. Some instructions can interrupt this sequence. GØ TØ means just what it says - go to the line named and start working from that point.

Follow this program as the computer does:

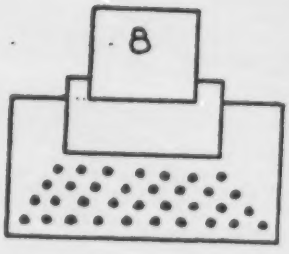


```
10 READ N,M
     .  | |
     .  | |
     .
50 DATA 5,3,6,9,7.1,....
```

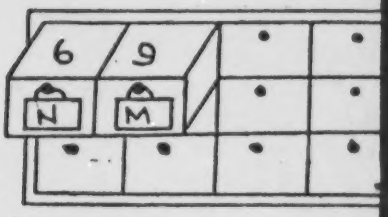
```
20 S = N + M
30 PRINT S
```



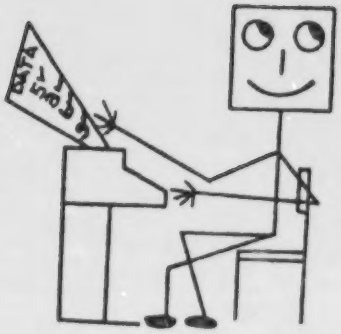

```
40 GØ TØ 10
```



At this point, the computer goes to instruction 10 and starts over again.



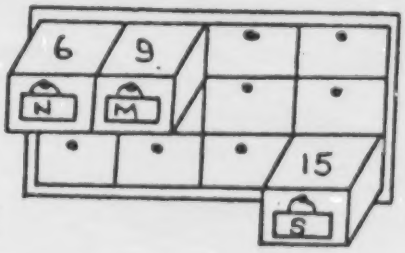

```
10 READ N,M
    .
    .
    .
50 DATA 5,3,6,9,....
```

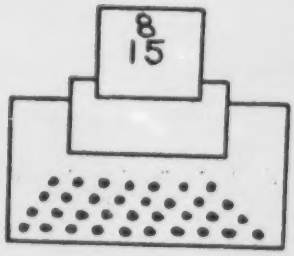


The computer keeps track of which numbers in the data statement have been used. Imagine that it is checking them off a list as they are read.

The contents of N and M are <u>replaced</u> by the next two numbers in the data statement.

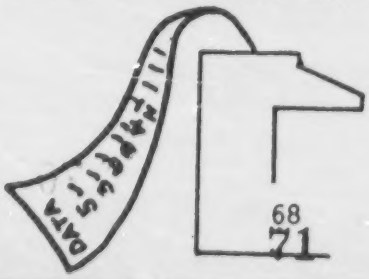
```
20 S = N + M
30 PRINT S
```

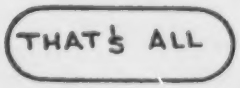




The old sum in S is <u>replaced</u> by the new sum.

When line 40 is reached, the computer again goes to line 10 and begins working from there. This time 4 and 7.1 replaces 6 and 9 in N and M, and 11.1 replaces 15 in S. The computer continues following these instructions until it runs out of numbers in the DATA statement.





Social Security taxes for household employees are paid by their employers every three months. Write a program which finds the total wages for each three-month period in a year. The data statement should contain twelve numbers, the wages for each month in the year.

Social Security tax is 8-1/4% of the total wages. Add a step to your program to calculate the tax for each quarter (three-month period.) This can be done by multiplying the three-month total by .0825.

Program written? O.K., how is this set of instructions communicated to the computer?

Input to computers can be accomplished in many different ways, but time-sharing uses just one simple method - programs are input by typing them on a teletype connected by telephone line to your time-sharing computer. All communications, that is, programs and system commands, are handled this way.

When your program is written, go over it carefully to be sure there are no mistakes, like left-out commas in DATA statements, or wrong line numbers in GØ TØ statements. The corrected program is then ready to be punched on paper tape. This is a simple process which records on tape everything you want to type to the computer. When the tape is run through the reader, it causes the teletype to type again exactly what you typed, but much faster.

PUNCHING PAPER TAPE

1. On the lower right hand side of the teletype is a switch. Turn it to LOCAL. This turns the teletype on, but does not connect it to the computer. The teletype is now ready to work just like an electric typewriter.

2. On the left of the teletype are two small pieces of equipment. The top one is a tape punch. Punch the button marked ON.

3. The bottom on the left is the tape reader. It's switch has three (or perhaps five) settings. Be sure it is set to FREE. Follow the instructions below to cut a tape of this program:

```
10 READ A, B, C
20 W = A + B + C
30 T = W*.0825
40 PRINT W, T
50 GO TO 10
60 DATA 90, 87, 25, 62, 15, 150, 89.50
70 DATA 175.25, 180, 52.25, 122, 30, 125.85
80 END
```

Hold down two keys on the teletype at one time - the keys for BREAK and repeat (REPT). Hold them down until about two inches of tape have come through the punch. This piece of tape is nearly whole, and is used for writing your name, or the name of the program, or whatever you want so that you will be able to identify the tape.

Hold down another two keys at one time - RUB OUT and REPT. The rub-out key causes the punch to punch all eight holes across the tape. About an inch of rubouts is enough to identify the start of the program and give you some header for placing the tape in the reader.

Now type the program exactly the way it is written. At the end of each line press the RETURN key, then the LINE FEED key, and then press the RUB OUT key at least three times. For example, to punch the first line of the social security program above, type:

10 READ A, B, C

Be sure to follow this sequence at the end of each line. After the program is typed, punch about two inches of rub-outs. Tear off the punched tape. Be careful when you tear the tape so that you will get the pointed arrow on the end. This arrow will help you when you are ready to place the tape in the tape reader.

CORRECTING MISTAKES

If you notice a mistake right after it is typed, it can be corrected with the back-arrow (←) key. Suppose a mistake has been made in typing PRINT

If you had made the mistake

```
PRIM
```

and noticed it immediately, you could correct the error by typing the SHIFT key and the Ø key at the same time. This will give you the back-arrow.

```
PRIM ←
```

The back-arrow indicates to the computer that the last character typed should be ignored.

```
PRIM← NT
```

would be read as PRINT by the computer.

If the mistake is not noticed when it is first made, it can still be corrected.

```
40 PRINT W,
```

could be corrected

```
40 PRINT W,←←←←← NT W, T
```

Each back-arrow "erases" one character. (A space is counted as a character.) This line would be interpreted by the computer as:

```
40 PRINT W, T
```

If you do not notice the error until the next line is typed, retype the line with the correct line number at the end of the program. The computer will automatically replace the incorrect line with the new line.

ENTERING AND RUNNING PROGRAMS FROM PAPER TAPE

1. Put tape reader switch on FREE. Open gate on reader by pressing down on catch. Put tape on reader so that small holes engage cogwheel with the three-punch side to the left and the five-punch side to the right. Be sure the "rub-out" part of the beginning of the tape is over the rectangular holes in the reader.

   Close gate over tape.

2. LOG-ON in regular manner. Don't forget to indicate language being used.

3. The computer needs to know if you want to work with a program stored in its memory or to enter a new program. It will type

   OLD, NEW, OR LIB FILE:

   You type NEW after the semicolon. Be sure to hit the RETURN key to send your message to the computer.

   The computer will also ask you to give your program a name. It will type

   FILE NAME:

   and wait for you to enter a name (the name may be up to eight characters long, only alphabetic or numeric characters may be used and the name must begin with an alphabetic character).

   After the semicolon you type the name of your program. Be sure to hit the RETURN key to send the message to the computer.

4. The computer will let you know when it is ready to proceed. It will type

   READY

   and will skip two lines and wait for you to begin.

5. In order for the computer to accept your paper tape, you must tell the computer that you are entering the program on tape. Type

   TAPE (Don't forget the RETURN key.)

   The computer will respond by typing READY and skipping two lines again.

6. Push the tape reader switch to START. The tape will pass through the reader, the teletype will copy each line of what is on the tape. It should print it out exactly as you had typed it earlier. The whole tape will be entered.

7. After all the lines have been entered into the computer, turn the tape reader to FREE. You must now let the computer know that the tape is finished. In order to return the computer to its normal mode of operation type

    NORMAL

    This command tells the computer that the tape is finished and that you wish to enter commands from the keyboard. The computer will again respond by typing

    READY

    The computer is waiting for you to tell it what you want done with the program you have just entered.

    After it has skipped two lines and returned to the margin you could type

    RUN

    which tells the computer to do what the program says. The computer will then follow all the commands you have given it and print out any information requested in the program.

    In place of typing RUN you could type

    LIST

    which will cause the computer to type the program back to you. This is done if you want to see if the computer received the message correctly.

8. Now the program may be completed, corrected, or saved in the manner as any program entered directly from the keyboard.

Name _____ Supplement A12.4

Period _____

SAMPLE TEST ON TOPIC 2

Indicate which operation is to be performed first: (assume a line #)

_____ 1. LET X = A + B * C

_____ 2. LET Y = C + 2 - (A + B)/2

_____ 3. LET V1 = C * D - Q/R

_____ 4. LET A1 = 3.14 * R + 2

_____ 5. C = X + Y - Z

Correct all errors in the following BASIC statements (if any):

_____ 6. 10 LET X + Y = Q

_____ 7. END

_____ 8. 45 GO TO TEN

_____ 9. 50 PRINT X "= x"

_____ 10. 20 LET D = (X + S/4

_____ 11. 70 LET A = PI * R + 2

_____ 12. 50 DATA 6, 8; 10, 12

Write the following formulas in assignment statements:

_____ 13. $x = r + \frac{5}{5}$

_____ 14. $x = \frac{2(b + c)}{a^2}$

_____ 15. $q = \frac{x^2 + y^2}{x + y}$

_____ 16. $y = 3x^2 - 2x - 4$

Determine the value of each BASIC variable at the completion of the sequence of steps below.

```
10 LET X = X + 10
20 LET Z = X + 1
30 LET W = 2 * X - Z
```

_____ 17. X

_____ 18. Z

_____ 19. W

77

Tell which of the following is a legal BASIC variable:

_____ 20. 2 * J

_____ 21. A1

_____ 22. BB

_____ 23. Z.4

If x = 1, y = 2, and z = 3, what value is assigned to W by each of these LET statements?

_____ 24. 10 LET W = X * Y + 2

_____ 25. 30 LET W = X/2 * Y + Z

_____ 26. 60 LET W = Z /(X + Y)

Evaluate:

_____ 27. Q =4 + 2 * 3 + 2

_____ 28. Y = 3 * 2 + 5 - 6/3

_____ 29. A = (4 + 8) * (5 - 3)

_____ 30. What value will be assigned to X by the following READ and DATA statements?

```
10 READ X, Y, Z
20 DATA 78, 34, 205
```

Write BASIC statments to do each of the following:

_____ 31. Assign to X the square root of P.

_____ 32. Multiply the contents of locations X and Y and store the product in location C.

_____ 33. Print the contents of X preceded by the message SLOPE =

Name _____ ANSWER KEY Supplement A12.4

Period _____

SAMPLE TEST ON TOPIC 2

Indicate which operation is to be performed first: (assume a line #)

* 1. LET X = A + B * C

\+ 2. LET Y = C ↑ 2 - (A + B)/2

* 3. LET V1 = C * D - Q/R

↑ 4. LET A1 = 3.14 * R ↑ 2

\+ 5. C = X + Y - Z

Correct all errors in the following BASIC statements (if any):

10 LET Q = X + Y — 6. 10 LET X + Y = Q

missing line # 100 END — 7. END

45 GO TO 10 — 8. 45 GO TO TEN

OK — 9. 50 PRINT X "= x"

missing parenthesis 20 LET D=(X+S/4) — 10. 20 LET D = (X + S/4

OK — 11. 70 LET A = PI * R ↑ 2

50 DATA, 6, 8, 10, 12 — 12. 50 DATA 6, 8; 10, 12

Write the following formulas in assignment statements:

X = R + S/5 — 13. $x = r + \frac{5}{5}$

X = 2 * (B + C)/A ↑ 2 — 14. $x = \frac{2\ (b + c)}{a^2}$

Q = (X ↑ 2 + Y ↑ 2)/(X + Y) — 15. $q = \frac{x^2 + y^2}{x + y}$

Y = 3 * X ↑ 2 - 2 * X - 4 — 16. $y = 3x^2 - 2x - 4$

Determine the value of each BASIC variable at the completion of the sequence of steps below.

```
10 LET X = X + 10
20 LET Z = X + 1
30 LET W = 2 * X - Z
```

10 — 17. X

11 — 18. Z

9 — 19. W

Tell which of the following is a legal BASIC variable:

No 20. 2 * J

Yes 21. A1

Yes 22. BB

No 23. Z.4

If x = 1, y = 2, and z = 3, what value is assigned to W by each of these LET statements?

4 24. 10 LET W = X * Y + 2

4 25. 30 LET W = X/2 * Y + Z

1 26. 60 LET W = Z/(X + Y)

Evaluate:

22 27. Q =4 + 2 * 3 + 2

9 28. Y = 3 * 2 + 5 - 6/3

24 29. A = (4 + 8) * (5 - 3)

78 30. What value will be assigned to x by the following READ and DATA statements?

```
10 READ X, Y, Z
20 DATA 78, 34, 205
```

Write BASIC statments to do each of the following:

LET X = SQR (P) 31. Assign to X the square root of P.

LET C = X * Y 32. Multiply the contents of locations X and Y and store the product in location C.

PRINT "SLOPE =" X 33. Print the contents of X preceded by the message SLOPE =

80

TOPIC THREE: DATA IN AND DATA OUT

Performance Objectives:

The student will -

A12 - Write programs which utilize INPUT, READ-DATA, PRINT, PRINT "-", commas, and semicolons.

References:

Textbook pages 67-100

Objective A12: The student will write programs which utilize INPUT, READ-DATA, PRINT, blank PRINT, commas, and semicolons.

Textbook Reference: Pp. 67-100

TEACHING SUGGESTIONS

Explain INPUT statement and its importance in allowing user participation. Contrast with use of LET.

READ-DATA statements need to be compared and contrasted to INPUT statements.

Discuss purpose of each and when each is most useful. Review floating point notation.

*Check the system used to determine whether double or single quotes are needed in PRINT statements.

*Also check system for method of spacing (commas, semicolons) and line skipping.

The number and length of print zones will vary from computer to computer. It is important to note, however, that alphameric output begins in the first space of the zone. Numerical output reserves the first space of the zone for a negative sign (if needed); otherwise, it is blank. Note that trying to print more expressions than zones (per line) causes the "extra" expressions to drop down to the next line.

Study the "packing" of output by the semicolon. Try the following sample statements--note all similarities and differences:

```
a)  PRINT A; B; C
b)  PRINT "A"; "B"; "C"
c)  PRINT "THE NUMBER IS"; X (try x=2 and x=-3)
d)  PRINT X; "IS THE NUMBER" (try x=2)
e)  PRINT "PERIMETER =" P
    PRINT "PERIMETER="; P
```

*Check the system against the information given at the bottom of p. 69 on the "? signal". Chances are there will be some variations.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral

Exercise One, pp. 70-71, All
Exercise Two, p. 77, All

Written

Problem Set Three, pp. 97-99, #1, 3, 4, 5, 7, 10, 11

Oral

Exercise Three, pp. 84-85, #1, 2, 3

Written

Problem Set Three, pp. 97-100, #2, 6, 12

Programs # 14 or 15, 16 or 17, 18, 19, 20, or 21

Supplement 12.4

System Spacing Test for Students

System Spacing Test for Students

These exercises will enable you to determine how your computer works to space or skip lines.

1) Enter and run this program to check for spacing.

```
5 PRINT "SPACING TEST"
10 READ A, B, C, D
20 DATA 5, -5, 0, 10
30 PRINT "SEMICOLONS"
35 PRINT A; B; C; D; A; B; C: D
40 PRINT "COMMAS"
45 PRINT A, B, C, D, A, B, C, D
50 PRINT "MIXTURE OF SEMICOLONS AND COMMAS"
55 PRINT A, B; C; D; A, B, C; D

60 PRINT "SEPARATE PRINTS"
65 PRINT A
66 PRINT B
67 PRINT C
68 PRINT D

70 PRINT "SEPARATE PRINTS WITH SEMICOLONS"
75 PRINT A;
76 PRINT B;
77 PRINT C;
78 PRINT D

80 PRINT "SEPARATE PRINTS WITH COMMAS"
85 PRINT A,
86 PRINT B,
87 PRINT C,
88 PRINT D
```

```
 90 PRINT "SEPARATE PRINTS WITH MIXED PUNCTUATION"
 95 PRINT A,
 96 PRINT B,
 97 PRINT C,
 98 PRINT D
100 END
```

2) Enter and run this program to check for line skipping.

```
 5 PRINT "LINE SKIPPING TEST"
10 READ A, B, C, D
15 DATA 5, -5, 0, 10
20 PRINT "BLANK PRINT"
22 PRINT A
23 PRINT
24 PRINT B
25 PRINT
26 PRINT
27 PRINT C
28 PRINT D

30 PRINT "BLANK PRINT AND COMMAS"
32 PRINT A,
33 PRINT
34 PRINT B,
35 PRINT ,
36 PRINT C,
37 PRINT ,
38 PRINT D
```

Name _____

Period _____

SAMPLE TEST ON TOPIC 3

30 pts.
Part I. Which of these are incorrect BASIC statements? Write corrected statements in the blanks provided.

1. 50 INPUT A; B _____
2. 20 DATA $2.35, $4.75 _____
3. 70 PRINT "INTEREST" = I _____
4. 60 PRINT "RADIUS OF THE SPHERE ="; R "CENTIMETERS"

   _____
5. 100 PRINT A B C _____
6. 10 INPUT 300 _____
7. 30 READ A + B _____
8. 20 PRINT 3 + 4 _____
9. 50 DATA 2,000; 3,500 _____
10. 40 READ x "IS A FACTOR OF" y _____

24 pts.
PART II.

A. Write each of the following numbers below in standard form.

1. 4.06077E + 09 _____
2. 1.00000E - 02 _____
3. 3.97979E + 08 _____
4. -2.00000E - 03 _____
5. 1.23455E + 06 _____

B. Write a print statment to perform the indicated arithmetic operations and print the result.

6. $\frac{15 + 17 - 13}{2}$ _____
7. $\frac{6.875}{3 \times 1.2}$ _____
8. $15^2 \times .032$ _____

24 pts.
PART III. Find any bugs (either syntax or programming errors). Show what will be printed if the corrected programs are run.

A.
```
IC REM PROGRAM TO COMPUTE
20 REM SIX PER CENT OF $23,000
30 LET D = 23,000
40 LET R = 6
50 LET R * D = A
60 PRINT "ANSWER IS"; A
70 END
```

OUTPUT _____

B.
```
10 REM PROGRAM TO AVERAGE
20 REM TWO NUMBERS
30 LET N1 = 24
40 LET N2 = 15
50 LET A = N1 + N2/2
60 PRINT AVERAGE IS, A
70 END
```

OUTPUT _____

20 pts.
Part IV. Write a short program so that it will perform the following task:

Evaluate the expression $i = j\ (k^2 + 2)$

if $j = 100$ and $k = 3$.

Use a single READ and DATA statement for data entry.

ANSWER KEY

SAMPLE TEST ON TOPIC 3

30 pts.
Part I. Which of these are incorrect BASIC statements? Write corrected statements in the blanks provided...

1. 50 INPUT A; B — 50 INPUT A, B
2. 20 DATA $2.35, $4.75 — 20 DATA 2.35, 4.75
3. 70 PRINT "INTEREST" = I — 70 PRINT "INTEREST = " I
4. 60 PRINT "RADIUS OF THE SPHERE ="; R "CENTIMETERS" — Correct
5. 100 PRINT A B C — Print A, B, C
6. 10 INPUT 300 — 10 INPUT A
7. 30 READ A + B — 30 READ A, B
8. 20 PRINT 3 + 4 — Correct
9. 50 DATA 2,000; 3,500 — 50 DATA 2,000, 3,500
10. 40 READ x "IS A FACTOR OF" y — 40 PRINT X " IS A FACTOR OF" y

24 pts.
PART II

A. Write each of the following numbers below in standard form.

1. 4.06077E + 09 — 4,060,770,000.
2. 1.00000E - 02 — .01
3. 3.97979E + 08 — 397,979,000
4. -2.00000E - 03 — -.002
5. 1.23455E + 06 — 1,234,550

B. Write a print statment to perform the indicated arithmetic operations and print the result.

6. $\frac{15 + 17 - 13}{2}$ — PRINT (15 + 17 - 13) /2
7. $\frac{6.875}{3 \times 1.2}$ — PRINT 6.875 / (3 * 1.2)
8. $15^2 \times .032$ — PRINT 15 ↑ 2 * .032

24 pts.
PART III. Find any bugs (either syntax or programming errors). Show what will be printed if the corrected programs are run.

A.
```
IC REM PROGRAM TO COMPUTE
20 REM SIX PER CENT OF $23,000
30 LET D = 23,000
40 LET R = 6
50 LET R * D = A
60 PRINT "ANSWER IS"; A
70 END
```

change IC to 10

```
LET D = 23000
LET R = .06
A = R * D
```

OUTPUT _____ 1380 _____

B.
```
10 REM PROGRAM TO AVERAGE
20 REM TWO NUMBERS
30 LET N1 = 24
40 LET N2 = 15
50 LET A = N1 + N2/2
60 PRINT AVERAGE IS, A
70 END
```

```
A = (N1 + N2)/2
"AVERAGE IS"; A
```

OUTPUT _____ Average is 19.5 _____

20 pts.
Part IV. Write a short program so that it will perform the following task:

Evaluate the expression $i = j\ (k^2 + 2)$

if j = 100 and k = 3.

Use a single READ and DATA statement for data entry.

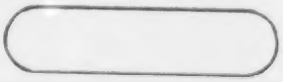
```
10 READ J, K

20 DATA 100, 3

30 LET I = J * (K ↑ 2 + 2)

40 PRINT I

50 END
```

TOPIC FOUR: FLOWCHARTS AND PROGRAM BRANCHING

Performance Objectives:

The student will -

A6 - Flowchart simple problems using standard symbols for

terminal 

assignment

INPUT/OUTPUT

decision

connector

A8 - Write BASIC programs from flowcharts.

A9 - Make a trace of a BASIC program.

A12 - Write and flowchart BASIC programs which utilize the following: READ-DATA, INPUT, REM STOP, END, PRINT, LET, GO TO, IF THEN and relational operators.

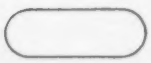
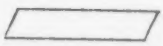
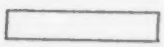
A16 - Produce a completely documented program.

References:

Textbook: pp. 101-136

Objective A6: Flowchart simple problems using standard symbols.

Textbook Reference: Topic 4, pp. 101-136

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Often textbooks begin flowcharting at an earlier stage than this book. Some teachers may wish to illustrate flowcharting from the beginning because once a student has had success with writing a program, he is often reluctant to do the preliminary work of flowcharting. Supplement 6.1 includes 2 flow charts which may be worked out in a class discussion or an individual assignment. (Perhaps, one can be done in class, assign the other.) Supplement 6.2 is an exercise to help students remember the symbols and form of a flowchart. Flowcharting should be individual, but point out ways that students can simplify their flowcharts. Paths should not overlap in most instances. The textbook omits the separate symbol for INPUT/ OUTPUT . The most common set of symbols is connector  terminal decision  input/output  assignment/computation flow of problem	Supplement 6.1 Supplement 6.2 Textbook Oral Exercise One, p. 106, #1-3 Written Problem Set Four, p. 132, #1:add e) input/output #3, 4 Supplement A6.2

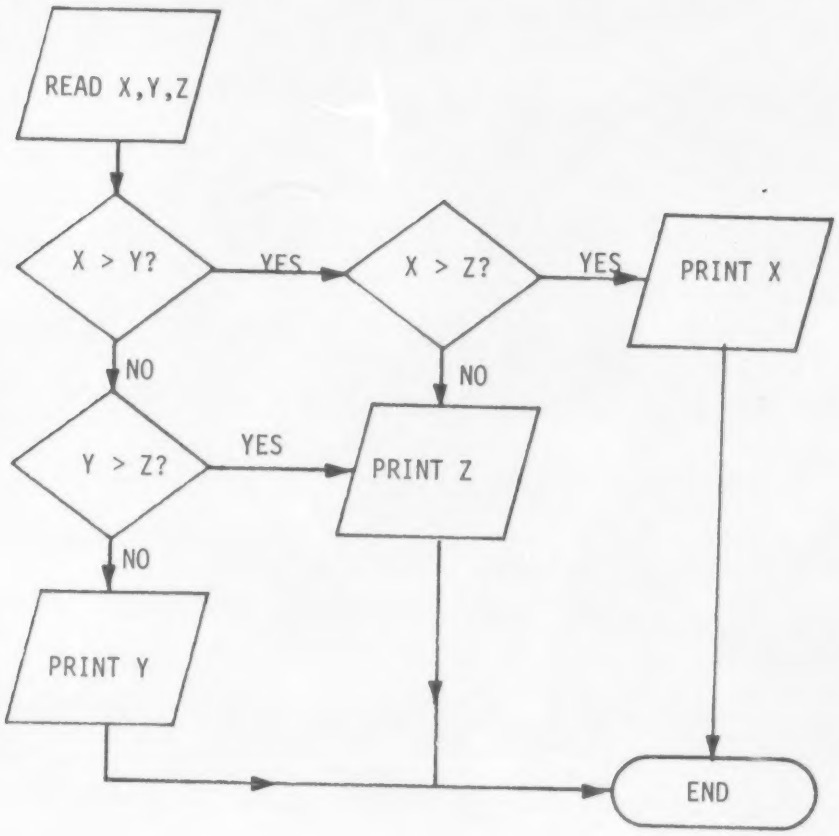
Flowcharting Exercises I

1) Read 3 numbers. Decide which number is largest by comparing only 2 at a time.
(Assume that no 2 numbers are equal.)

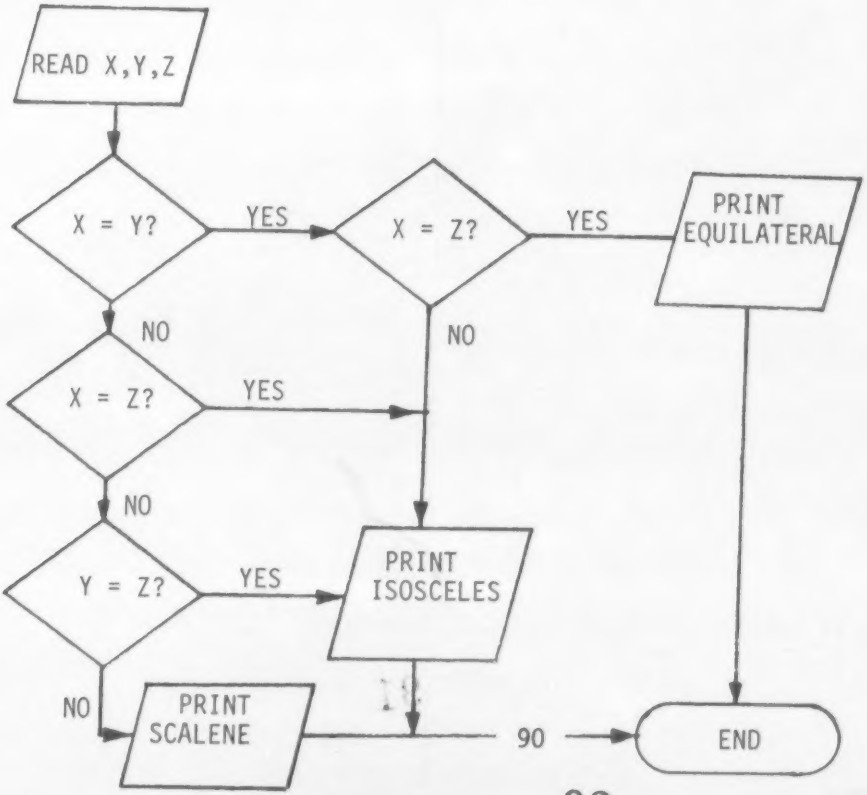
2) Read the 2 sides of a triangle, X, Y, and Z. Decide if the triangle is scalene, isosceles, or equilateral.

Sample flowcharts:

1)



2)

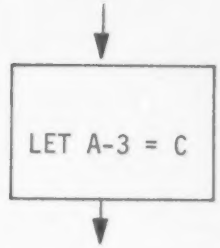


Flowcharting Exercise II

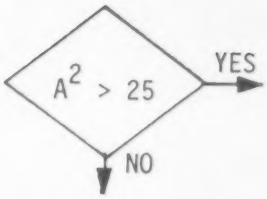
Supplement A6.2

Determine whether the following are valid flowchart boxes.
Make corrections, if they are invalid.

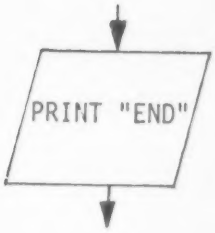
1)



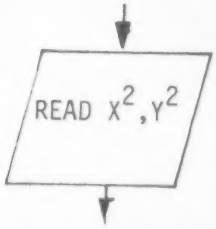
2)



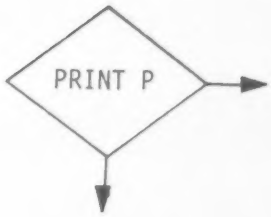
3)



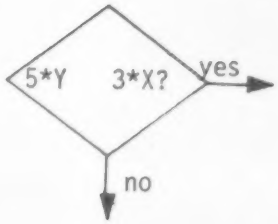
4)



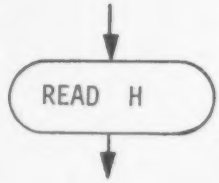
5)



6)

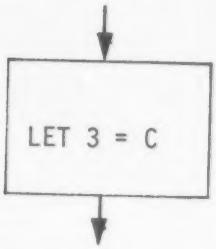


7)

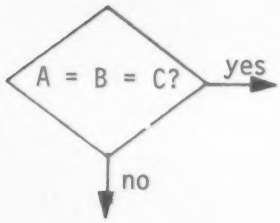


8)

9)



10)



Draw a flowchart to:

11) Assign to location B, three times the number in location A

12) Test if A times B is greater than C times D.

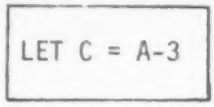
13) Print, with a suitable message, that the perimeter is P.

Draw a portion of the flowchart to fit each problem:

14) If $x^2 = 9$ is true, print X "IS A ROOT"; if it is false, END.

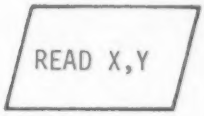
15) Decide whether A + B = 6 is true. If it is, print "YOU WIN!"; if it does not, test whether A + B = 8 is true.

ANSWERS TO SUPPLEMENT A6.2

1) no - change to 

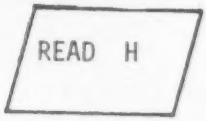
2) yes

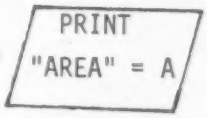
3) yes

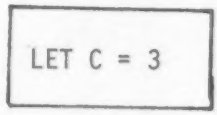
4) no - change to 

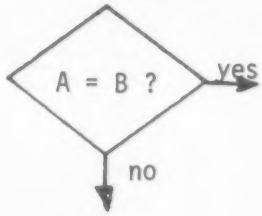
5) no - change to

6) yes

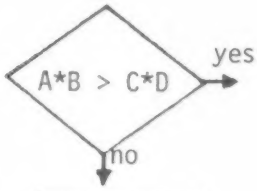
7) no - change to 

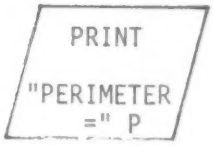
8) no - change to 

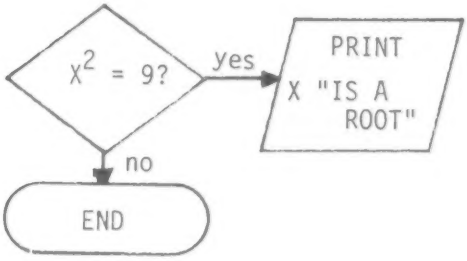
9) no - change to 

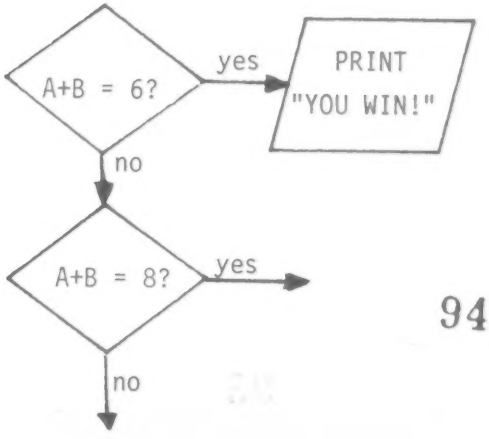
10) no = change to 

11)

12) 

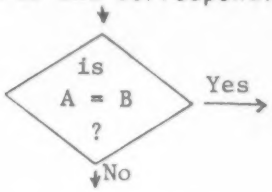
13) 

14) 

15) 

Objective A8: The student will write BASIC programs from flowcharts.

Textbook Reference: Pp. 108-115

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Go over carefully examples of decisions in flowcharts and corresponding BASIC statements.  30 if A = B THEN GO Clarify any differnce between normal algebraic relation symbols and BASIC decision symbols available on unit. NOTE: Remind students of input/output symbol not given in textbook. NOTE: P. 112 - This discussion of the use of STOP may be confusing to students. The GO TO statement can be explained here, rather than waiting until p. 116.	Textbook Oral Objective Activity pp. 126-128, #2, 6 Written Problem Set Four, p. 133, #2, 6

Objective A9: The student will make a trace of a BASIC program.

Textbook Reference: Page 115

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Trace steps in BASIC programs. This is especially useful in preparing students for Debugging in more sophisticated programs. See Supplement 9.1 for sample trace. Supplement 9.2 may be reproduced as a form for students to use with their programs.	Textbook Oral Exercise Two, p. 115, #2, 3 Oral Activity, p. 128, #5

TRACING A PROGRAM

Sample Program:

```
10 LET S = D
20 READ L, W
30 LET A = L * W
40 PRINT A
50 LET S = S + 1
60 IF S < 4 THEN 20
70 DATA 2, 4, 3, 5, 6, 8, 1, 10
80 END
```

Sample Trace:

Statement #	Variables				Printout	Decision
	S	L	W	A		
10	0					
20		2	4			
30				8		
40					8	
50	1					
60						S < 4
20		3	5			
30				15		
40					15	
50	2					
60						S < 4
20		6	8			
30				48		
40					48	
50	3					
60						S < 4
20		1	10			
30				10		
40					10	
50	4					
60						S < 4
80						END

Supplement A9.2

Trace Sheet

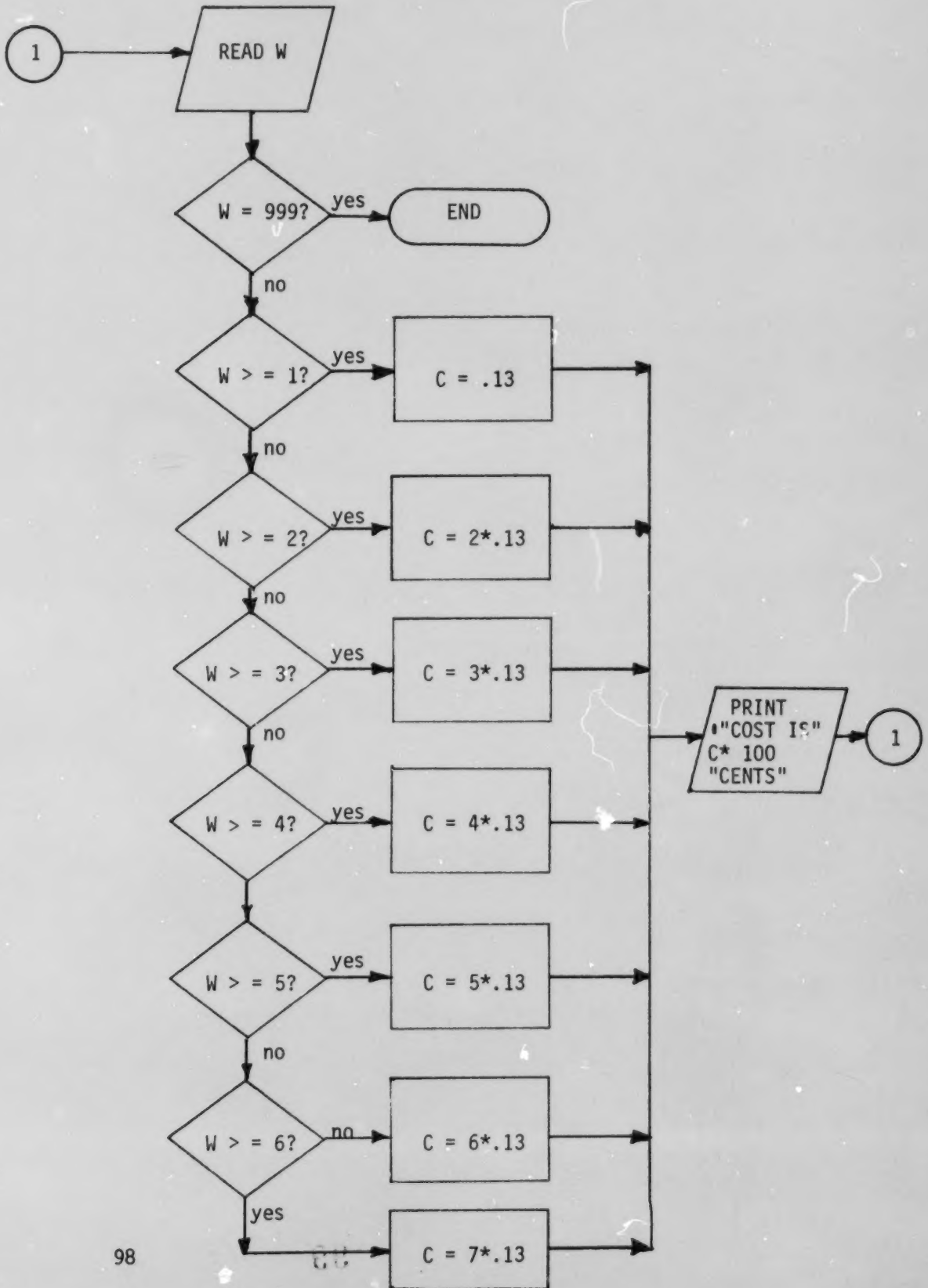
Statement #	Variables	Printout	Decision

Objective A12: The student will write and flow chart BASIC programs which utilize READ-DATA, INPUT, REM, PRINT, END, LET, GO TO, IF-THEN, and relational operators.

Textbook Reference: Pp. 116-119

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Go over use and format of GO TO. GO TO statements are used to BRANCH in a) to skip ahead b) to loop in a program Format of GO TO statement 40 GO TO 90 Line number following GO TO must be an executable statement and must not branch to itself, if not in the graphics mode. 99 GØ TØ 99 is used in graphics to keep the picture on the screen. Clarify distinction between GO TO and IF-THEN 1) use of IF-THEN and trailer data to prevent infinite loops 2) conditional (IF-THEN) and unconditional (GO TO) branching Continue to develop trace skills.	Textbook Oral Exercise Three, p. 122, #1 Written Problem Set Four, p. 134, #8 P. 48 in the Teacher's Guide gives an incorrect solution to #17, p. 136. A replacement with corrections is included in this guide as Supplements A12.1 and A12.2. Textbook Oral Exercise Three, pages 122-123, #2, 3 Objective Activity, pp. 127-129, #3-8 Written Problem Set Four, pp. 133-134, #5, 7, 9-10 PROGRAMS Problem Set Four, pp. 135-136, #11-18

Flowchart #17, p. 136



```
10 REM #17 PAGE 136
20 REM CALCULATING POSTAGE RATES
30 READ W
40 IF W = 999 THEN 270
50 IF W>=1 THEN 80
60 C=.13
70 GOTO 240
80 IF W>=2 THEN 110
90 C=2*.13
100 GOTO 240
110 IF W>=3 THEN 140
120 C=3*.13
130 GOTO 240
140 IF W>=4 THEN 170
150 C=4*.13
160 GOTO 240
170 IF W>=5 THEN 200
180 C=5*.13
190 GOTO 240
200 IF W>=6 THEN 230
210 C=6*.13
220 GOTO 240
230 C=7*.13
240 PRINT "COST IS" C*100 "CENTS"
250 GOTO 30
260 DATA 0.47,1.71,3.54,6.32,999
270 END
```

```
COST IS 13 CENTS
COST IS 26 CENTS
COST IS 52 CENTS
COST IS 91 CENTS
```

Objective A16: The student will produce a fully-documented program for a term project.

Textbook Reference:

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
At this point students need to put together skills already developed to prepare a fully documented program. Supplement 16.1 is one method that can be used. This is a good time to assign the term project which should be a completely documented program of some complexity. The programs from Objectives 17 and 17a are suitable for term projects. In all probability, students will resist the documentation of programs; once their programs run correctly, they consider them complete. Stress the need for documentation as a guide to users of their programs. Point out that documentation is required for all professional programming; the ability to document is therefore a skill to be developed. All major projects should be documented. The exact form of documentation is up to the individual teacher. The most important part of documentation is in-source documentation, that which is done through remarks in the program itself. Supplement 16.2 can be used as a guide, but teachers may not wish to require all of the documentation in it.	Begin term project. Supplement A16.1 Supplement A16.2

PROGRAM DOCUMENTATION

I. Statement of problem

II. Description of method

Include a list of variables and how they are used.

III. Flow Chart

IV. Program (LIST)

Program should be past point of compilation errors. It should contain heading and remark statements. Use as many remark statements as needed for program readability.

V. Output (RUN)

RUN should be correctand in format that will be clear to a person not using the computer.

VI. Evaluation

What does the program do?

What are its good and bad points?

Is it general enough to have multiple uses?

Any other comments?

SAMPLE DOCUMENTED PROGRAM

Heading:

John Doe
December 5, 1980
Salary and Withholding Program

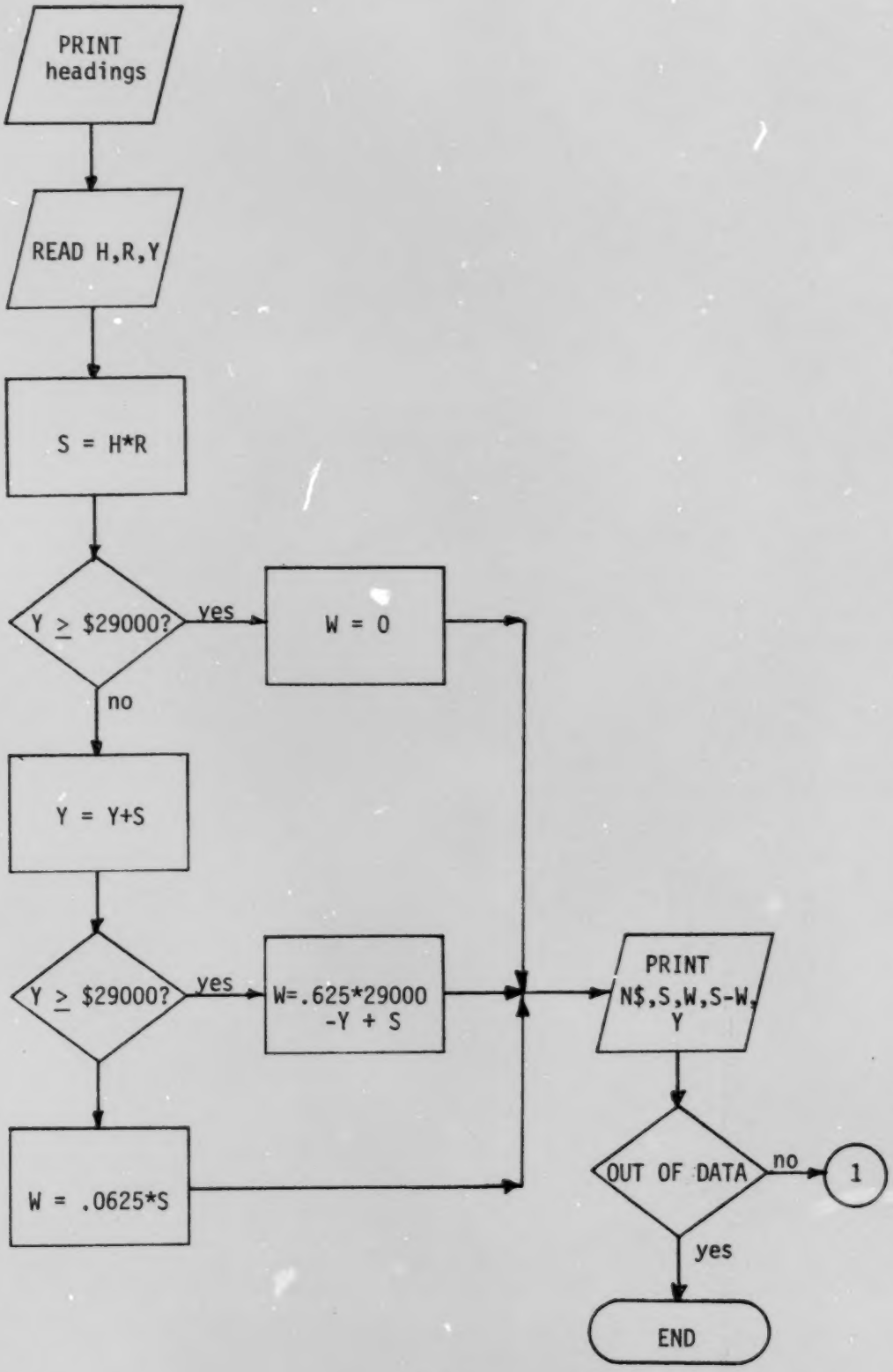
Statement of the Problem:

This program calculates a worker's weekly pay, the Social Security withheld, the net pay, and the year-to-date earnings. Input consists of the worker's name, the number of hours worked during the week, the hourly pay rate, and the year-to-date earnings prior to the week. A constant factor is that Social Security is withheld at a rate of 6.25% on the first $29,000 earned in a calendar year.

Description of Method:

The weekly salary (S) is computed by multiplying the number of hours worked (H) by the hourly pay rate (R). If the year-to-date earnings (Y) are equal or exceed $29,000, there is no withholding (W), and the net pay (S-W) is the same as the salary (S). If the year-to-date pay (Y) is less than $29,000, the weekly salary (S) is added to it, and it is again tested to see if it exceeds $29,000. If it (Y) does, 6.25% of the difference ($29,000 - Y + S) is calculated. If the year-to-date (Y) is still less than $29,000 the withholding (W) is found my multiplying 6.25% by the weekly salary (S). The net pay is salary (S) minus withholding (W). The new year-to-date figure (Y) is the old amount plus the weekly salary (S).

Flowchart:


PRINT headings
READ H,R,Y
S = H*R
Y ≥ $29000?
yes
W = 0
no
Y = Y+S
Y ≥ $29000?
yes
W=.625*29000 -Y + S
W = .0625*S
PRINT N$,S,W,S-W, Y
OUT OF DATA
no
1
yes
END

```
10 REM JOHN DOE
20 REM DECEMBER 5,1980
30 REM SALARY AND WITHHOLDING PROGRAM
40 PRINT "EMPLOYEE"; TAB(16);"SALARY";TAB(26);"WITHHOLDING";
50 PRINT TAB(40);"NET PAY"; TAB(50);"YEAR-TO-DATE":PRINT
60 REM N$=EMPLOYEE NAME
70 REM H=HOURS WORKED THIS WEEK
80 REM R = HOURLY PAY RATE
90 REM Y = YEAR-TO-DATE EARNINGS
95 A$="$$######.##"
100 READ N$,H,R,Y
110 REM S = WEEKLY SALARY
120 S=H*R
130 REM SOCIAL SECURITY IS WITHHELD AT A RATE OF 6.25% FOR THE
140 REM FIRST $29,000 EARNED IN THE CALENDAR YEAR
150 IF Y>= 29000 THEN 210
160 Y=Y+S
170 IF Y>=29000 THEN 250
180 REM W = WITHHOLDING ON WEEKLY SALARY
190 W=.0625*S
200 GOTO 270
210 W=0
220 REM WITHHOLDING IS ZERO IF EARNINGS OVER $29,000
230 Y = Y+S
241 GOTO 270
250 W = .0625*(29000-Y+S)
270 PRINT N$;TAB(14);USING A$;S;
280 PRINT TAB(25);USING A$;W;
290 PRINT TAB(38);USING A$;S-W;
300 PRINT TAB(50);USING A$;Y
310 GOTO 100
320 DATA "ADAMS,J.",40,30,28500
330 DATA "BAKER,S.",40,12.5,26500
340 DATA "COOK,E.",0,20,27000
350 DATA "DALY,M.",40,15,25000
360 DATA "JONES,G.",36,35,33000
370 END
```

EMPLOYEE	SALARY	WITHHOLDING	NET PAY	YEAR-TO-DATE
ADAMS,J.	$1200.00	$31.25	$1168.75	$29700.00
BAKER,S.	$500.00	$31.25	$468.75	$27000.00
COOK,E.	$0.00	$0.00	$0.00	$27000.00
DALY,M.	$600.00	$37.50	$562.50	$25600.00
JONES,G.	$1260.00	$0.00	$1260.00	$34260.00

Name_____

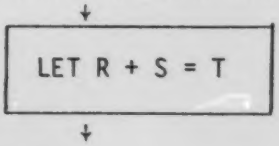
Period_____

SAMPLE

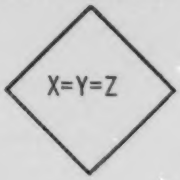
TEST TOPIC FOUR

Part I. (4 pts. each - Total 40 pts.)

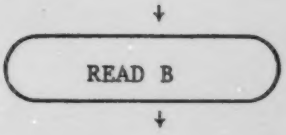
For each box, state whether it is a valid flowchart box. (Yes or No)

1. 

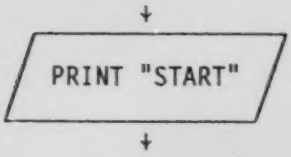
_____

2. 

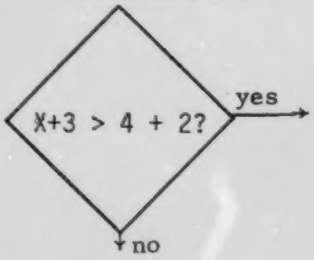
_____

3. 

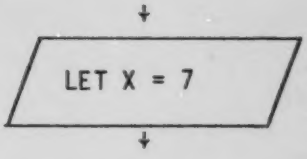
_____

4. 

_____

5. 

_____

6. 

_____

Draw a flowchart to execute each task.

7. Test of 6 * B = A.

8. Assign 3 plus X to Y.

9. Print the area of the triangle
   is the variable A.

10. Terminate the flowchart.

Part II. IF/THEN statements (20 pts.)

Identify the error and correct.

1. 20 IF A EQUALS B THEN GO _____

2. 50 IF (B = C) THEN 90 _____

3. 40 IF L > = T, THEN 20 _____

4. 10 IF M < B GO TO 100 _____

5. 70 IF A < > B THEN 70 _____

PART III.

Draw the flowchart and write the program for the following problem:

The formula for calculating distance is d = rt. Where d stands for distance, r stands for rate, and t stands for time. Read the given value for R and T and calculate D. Use the trailer number to terminate the program. Print the rate, time, and distance in a table.

Rate in Km/h	Time in hours
60	3
45	4.5
675	4.5
-99999	-99999

Name _____

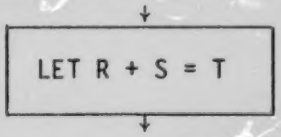
ANSWER KEY Period _____

SAMPLE

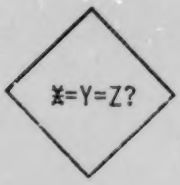
TEST TOPIC FOUR

Part I. (4 pts. each - Total 40 pts.)

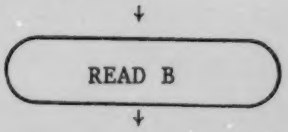
For each box, state whether it is a valid flowchart box. (Yes or No)

1. 

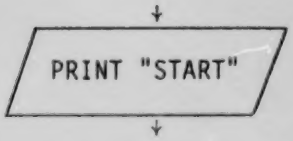
No

2. 

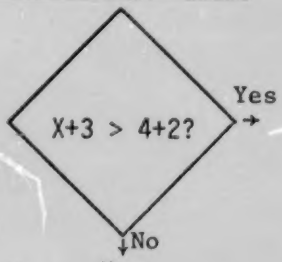
No

3. 

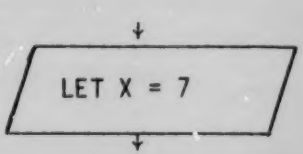
No

4. 

Yes

5. 

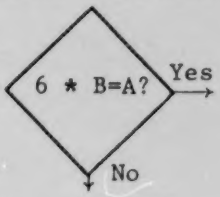
Yes

6. 

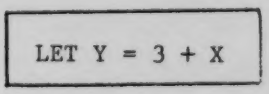
No

Draw a flowchart to execute each task.

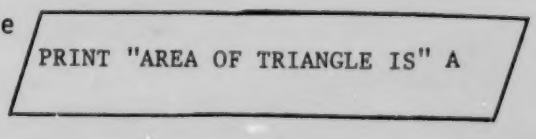
7. Test of 6 * B = A.



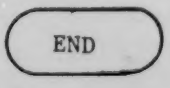
8. Assign 3 plus X to Y.



9. Print the area of the triangle is the variable A.



10. Terminate the flowchart.



Part II. IF/THEN statements (20 pts)

Identify the error and correct the statement.

1. 20 IF A EQUALS B THEN GO _____ 20 IF A = B THEN 80 _____

2. 50 IF (B = C) THEN 90 _____ 50 IF B = C THEN 90 _____

3. 40 IF L > = T, THEN 20 _____ 40 IF L > = T THEN 100 _____

4. 10 IF M < B GO TO 100 _____ 10 IF M < B THEN 100 _____

5. 70 IF A < > B THEN 70 _____ 70 IF A < > B THEN 90 _____

TOPIC FIVE: FOR-NEXT LOOPS

Performance Objective:

The student will -

A10 - Debug a BASIC program of ten or more statements.

A12 - Flow chart a solution to the problem and write and process a BASIC program which contains a FOR-NEXT loops.

References:

Textbook pp. 137-169; Appendix A, pp. 451-453

Objective A10: The student will debug a BASIC program of ten or more statements.

Textbook Reference: Pp. 451-453

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Debugging will be used in all work done. At this point it is important that students have a grasp of the methods used in debugging programming and logical errors. 1) Programming errors. These are mainly syntax errors. These errors are usually diagnosed by the computer. Check system used. 2) Logical errors. These are the most difficult errors to diagnose. Two methods often found useful are: a) a hand-trace b) the insertion of extra output statments throughout the program so that step-by-step-processes can be checked and the location of the error more easily determined.	Debugging is constantly done in all programs. Students need to be encouraged to rely on and develop their own debugging skills instead of seeking the teacher's assistance.

COMPUTER MATHEMATICS

Objective A12: FOR-NEXT Loops

Textbook Reference:

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
This topic is one with which students sometimes have initial problems. Use of careful traces is quite useful in helping students understand loops. Flowchart first, then program. First use IF THEN in simple counting the summation loops, then show efficiency of FOR-NEXT. At this point explain components of loops--comparing IF-THEN and FOR-NEXT in flowcharts and programs.	Textbook Oral Exercise One, p. 141, #1-3 Written Exercise Two, pp. 146-147, #1-3 Exercise Two, pp. 146-147, #1-3 Objective Activity, pp. 158-160, #1-7 Supplement 12.5 To check increment-test order of FOR-NEXT in system used.
Compare IF-THEN and FOR-NEXT loop flowcharts and programs. Again traces are useful. Errors in textbook P. 141 - Exercise One, #2 Discussion of problem on p. 142 is incorrect. As is, the program prints 5 indefintely. If line 60 is changed to GO TO 30, then 5 will print 3 times (as p. 142 says). P. 160 - #6a - is correct - not printed as is stated on p. 163 with mismatched indices. Errors in Teacher's Edition P. 52 - #9a should be 40 LET I=I + 2 #10 line 40 is omitted, should be 40 FOR J=I TO F STEP K P. 55 - #18 answers are incorrect. See attached program.	Textbook Oral Exercise Three, pp. 151-152, #1-5 Exercise Four, p. 157, #1-2 Objective Activity, pp. 160-161, #8-15 Written Problem Set Five, pp. 166-169, #6-14 Programs Problem Set Five, p. 169, #15-18

The looping mechanism on various systems may vary. Some do not compare the initial value of the index with the final value; one loop will be completed before a comparison is made. The final value of the index can vary also. In some systems after a loop FOR I = 1 TO 6... NEXT I has been completed, the value of I is 6, but in other systems I is 7. Being aware of these technicalities can eliminate potential programming problems. To point out these variations in looping mechanisms to the students, use Supplement 12.5.

```
10 REM PROBLEM 18 SET 5
20 PRINT "RADIUS","AREA"
25 LET A$="####.##"
30 LET PI = 3.14
40 FOR R = 35 TO 50
50 LET A = PI * R^2
60 PRINT R,
65 PRINT USING A$;A
70 NEXT R
80 END
```

```
RADIUS          AREA
 35             3846.50
 36             4069.44
 37             4298.66
 38             4534.16
 39             4775.94
 40             5024.00
 41             5278.34
 42             5538.96
 43             5805.86
 44             6079.04
 45             6358.50
 46             6644.24
 47             6936.26
 48             7234.56
 49             7539.14
 50             7850.00
```

CHECK OF LOOPING MECHANISM

Run these simple programs to test the looping mechanism of the computer.

```
10 N = 2
20 FOR I = 3 TO N
30 PRINT I
40 NEXT I
50 END
```

Is any value for I printed?

```
10 FOR I = 1 TO 6
20 NEXT I
30 PRINT I
90 END
```

What value of I is printed?

SAMPLE TEST

TOPIC FIVE

1. What is the output of this program?
(8 pts.) (Be careful.)

```
100 LET I = 1
110 READ X
120 IF X = 9999 THEN 180
130 PRINT X
140 LET I = I + 1
150 IF I < = 3 THEN 110
160 PRINT I
170 DATA 7, 3, 18, 21, 42, 9999
180 END
```

2. The following program does not do what it claims. Find and correct all errors.
(12 pts.)

```
20 I = 1
30 S = 0
40 S = S + I
50 I = I + 1
60 IF I > 10 THEN 30
70 PRINT I
80 END
```

3. Match the step #'s of the program with the letter corresponding to the proper component of this counting loop.
(12 pts.)

```
10 LET X = 5
20 LET N = 1
30 PRINT X
40 IF N > = 5 THEN 70
50 LET N = N + 3
60 GO TO 30
70 END
```

_____ a. test

_____ b. exit

_____ c. re-entry

_____ d. initial value

_____ e. increment

_____ f. What is the index?

4. What will be printed when these programs are run?
(16 pts.)

a.
```
10 FOR J = 2 to 4
20 PRINT J + 2
30 NEXT J
40 END
```

b.
```
10 FOR N = 5 to -3  STEP -2
20 PRINT N
30 PRINT N
40 END
```

Find one error in each of these programs. Circle it.

5. (5 pts.)

```
10 REM 6 % PROGRAM
20 FOR N = 1 TO 6
30 READ X
40 PRINT X, .06 * X
50 NEXT X
60 DATA 85, 90, 95, 70, 75, 85
70 END
```

6. (5 pts.)

```
10 REM COUNT THE
20 REM POSITIVE #'s
30 REM TYPED
35 FOR I = 1 TO 10
40 C = 0
50 INPUT N
60 IF N < = 0 THEN 80
70 LET C = C + 1
80 NEXT I
90 PRINT C" "ARE POSITIVE"
100 END
```

7. 45 FOR Y = 70 TO 135
(8 pts.)

Identify:
a) index variable _____
b) initial value _____
c) final value _____
d) increment value _____

8. Rewrite the following counting loop as a FOR-NEXT loop making the appropriate alterations.
(8 pts.)

```
30 LET I = 3
40 PRINT I
50 IF I > = 13 THEN 80
60 LET I = I + 2
70 GO TO 40
80 END
```

9. Each of the following sequence of statements is invalid. Circle the error in each.
(6 pts.)

a.
```
10 FOR J = 1 TO 6
20 LET J = J + 2
30 NEXT J
```

b.
```
10 FOR C = 1 TO 10 STEP 2
20 PRINT C ↑ 3
30 NEXT C
40 GO TO 20
```

10. a) Flowchart and b) write the program for the following problem:
(20 pts.)

Using a FOR-NEXT loop, print the squares of every tenth integer from 10 to 100.

ANSWER KEY
SAMPLE TEST

TOPIC FIVE

1. What is the output of this program?
(8 pts.) (Be careful.)

```
100 LET I = 1
110 READ X
120 IF X = 9999 THEN 180
130 PRINT X
140 LET I = I + 1
150 IF I < = 3 THEN 110
160 PRINT I
170 DATA 7, 3, 18, 21, 42, 9999
180 END
```

```
7
3
18
4
```

2. The following program does not do what it claims. Find and correct all errors.
(12 pts.)

```
20 I = 1
30 S = 0
40 S = S + I
50 I = I + 1 <---------------- Change to 2
60 IF I > 10 THEN 30 <---------- Change to 40;  Change > to <
70 PRINT I <-------- Change to S
80 END
```

3. Match the step #'s of the program with the letter corresponding to the proper component of this counting loop.
(12 pts.)

```
10 LET X = 5                    40   a. test
20 LET N = 1                    70   b. exit
30 PRINT X                      60   c. re-entry
40 IF N > = 5 THEN 70           20   d. initial value
50 LET N = N + 3                50   e. increment
60 GO TO 30                      N   f. What is the index?
70 END
```

4. What will be printed when these programs are run?
(16 pts.)

a.
```
10 FOR J = 2 to 4
20 PRINT J + 2          4
30 NEXT J               5
40 END                  6
```

b.
```
10 FOR N = 5 to -3  STEP -2
20 PRINT N          5
30 PRINT N          3
40 END              1
                   -1
                   -3
```

Find one error in each of these programs. Circle it.

5. (5 pts.)

```
10 REM 6 % PROGRAM
20 FOR N = 1 TO 6
30 READ X
40 PRINT X, .06 * X
50 NEXT (X) N
60 DATA 85, 90, 95, 70, 75, 85
70 END
```

6. (5 pts.)

```
10 REM COUNT THE
20 REM POSITIVE #'s
30 REM TYPED
35 FOR I = 1 TO 10
40 C = 0
50 INPUT N
60 IF N < = 0 THEN 80
70 LET C = C + 1
80 NEXT I
90 PRINT C" "ARE POSITIVE"
100 END
```

7. 45 FOR Y = 70 TO 135
(8 pts.)

Identify:

a)	index variable	Y
b)	initial value	70
c)	final value	135
d)	increment value	1

8. Rewrite the following counting loop as a FOR-NEXT loop making the appropriate alterations.
(8 pts.)

```
30 LET I = 3
40 PRINT I
50 IF I > = 13 THEN 80
60 LET I = I + 2
70 GO TO 40
80 END
```

```
10 FOR I = 3 TO 13 STEP 2
20 PRINT I
30 NEXT I
40 END
```

9. Each of the following sequence of statements is invalid. Circle the error in each.
(6 pts.)

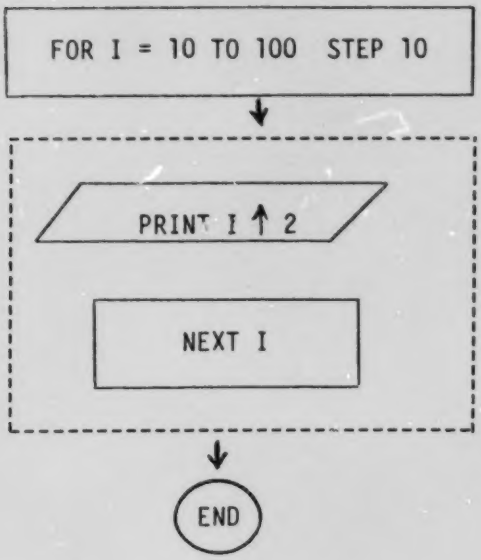
a.
```
10 FOR J = 1 TO 6
20 LET J = J + 2
30 NEXT J
```

b.
```
10 FOR C = 1 TO 10 STEP 2
20 PRINT C ↑ 3
30 NEXT C
40 GO TO 20
```

10. a) Flowchart and b) write the program for the following problem:
(20 pts.)

Using a FOR-NEXT loop, print the squares of every tenth integer from 10 to 100.

10. (continued)




```
10 FOR I = 10 TO 100 STEP 10
20 PRINT I ↑ 2
30 NEXT I
40 END
```

TOPIC SIX: ACCUMULATING AND TABLE HEADING

Performance Objectives

The student will -

A14 - Write BASIC programs utilizing TAB

A14a - Write BASIC programs utilizing the functions SPC and PRINT USING.

A17a - Prepare table of integers, squares and cubes.

References

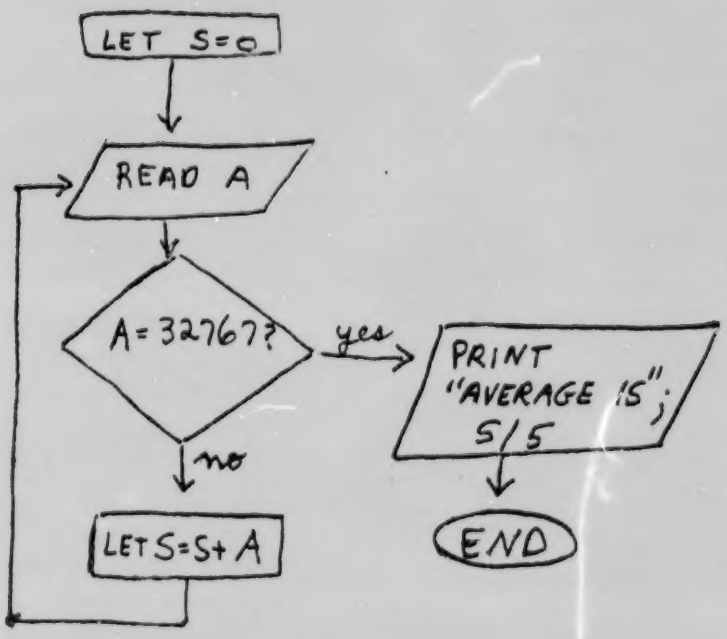
Textbook: Pp. 171-201

Objective A14: The student will write BASIC programs utilizing TAB.
A17a: The student will prepare a table of integers, squares and cubes.

Textbook Reference: Pages 171-201

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
The first part of this topic is mainly review and use of FOR-NEXT in finding sums and averages. Functional Output - The exact function of commas, semicolons, and TAB varies from system to system. The system used needs to be checked out by teacher or students. Error in Teacher's Guide - There is an error in the Teacher's Guide, p. 58, #10 (p. 201 - #10 in textbook). The program does not average the numbers as per instructions. The flowchart is also incorrect.	Textbook Oral Exercise Three, p. 190, #1-2 Objective Activity, pp. 194-196, #1-9 Written Problem Set Six, pp. 199-200, #1-6 Programs Problem Set Six, p. 201, #8-12 Supplement 14.1 - Print Projects Program Write a program to take the odd integers from one to twenty and print the integer, its square and its cube in a table with proper headings.
Some students may want to do artistic designs using such BASIC statements as PRINT, TAB, AND DEF FN. These print projects may be used for extra credit. Many examples of sophisticated computer art are in the textbook. Some machines count the first space as TAB(0); others use TAB(1). Check your machine.	Supplement A14

```
10 REM PROBLEM 10
20 LET S = 0
30 READ A
35 DATA 56,34.8,79.3,269,57.2,32767
40 IF A = 32767 THEN 70
50 LET S = S+A
60 GOTO 30
70 PRINT "AVERAGE IS  ";S/5
80 END


AVERAGE IS   99.26
```



COMPUTER DESIGNS

Printing designs on the computer can be more difficult than it seems at first glance; on many systems, blocks are rectangular, not square, and designs become distorted because of this.

An example of a simple design:
(Change PRINT TO LPRINT for a hard copy.)

```
 5 K=1
10 FOR I = -1 TO 9 STEP 2
20 PRINT TAB (33-K); STRING$ (I+2, "*")
30 K=K+1
40 NEXT I
45 K=K-1
50 FOR I = 9 TO -1 STEP-2
60 PRINT TAB (33-K); STRING$ (I+2, "*")
70 K=K-1
80 NEXT J
90 END
```

The same design may also be produced by this program:

```
 10 PRINT TAB(32) "*"
 20 PRINT TAB(31) "***"
 30 PRINT TAB(30) "*****"
 40 PRINT TAB(29) "*******"
 50 PRINT TAB(28) "*********"
 60 PRINT TAB(27) "***********"
 70 PRINT TAB(27) "***********"
 80 PRINT TAB(28) "*********"
 90 PRINT TAB(29) "*******"
100 PRINT TAB(30) "*****"
110 PRINT TAB(31) "***"
120 PRINT TAB(32) "*"
130 END
```

A third program is much less flexible and much more time-consuming.

```
10 PRINT "  (32 spaces)          *"
20 PRINT "  (31 spaces)         ***"
30 PRINT "  (30 spaces)        *****"
.
.
.
and so on.
```

Use loops and the STRING$ Command (if available on your system) in print programs wherever possible. The more elaborate the design, the more time it will save.

COMPUTER MATHEMATICS

OBJECTIVE A14a: The student will write BASIC programs utilizing the functions SPC and PRINT USING.

Textbook Reference:

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
The PRINT USING statement can make reports easier to read. Consult your computer manuals for exact specifications and syntax. The program #9 on p. 198 of the textbook can be used as an example: Make the following substitutions and/or additions.	

```
5 A$="$$#####.##"
10 REM OBJECTIVE ACTIVITY
20 REM PROBLEM 9
30 PRINT "EMPLOYEE NUMBER";TAB(20);"SALARY";TAB(36);"BONUS"
40 LET T = 0
50 FOR L = 1 TO 10
60 READ S
70 LET B=S/12
80 LET T = T+B
90 PRINT TAB(6);L;TAB(19);USING A$;S;
91 PRINT TAB(33);USING A$;B
100 NEXT L
110 PRINT
120 PRINT "TOTAL BONUS  ";
121 PRINT USING A$;T
130 DATA 10500,9850,11500,8840,9000
140 DATA 7700,15000,14560,25000,19000
150 END
```

```
EMPLOYEE NUMBER     SALARY          BONUS
      1            $10500.00       $875.00
      2             $9850.00       $820.83
      3            $11500.00       $958.33
      4             $8840.00       $736.67
      5             $9000.00       $750.00
      6             $7700.00       $641.67
      7            $15000.00      $1250.00
      8            $14560.00      $1213.33
      9            $25000.00      $2083.33
      10           $19000.00      $1583.33

TOTAL BONUS  $10912.50
```

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
The function SPC(X) - sometimes SPA(X) - causes X blank spaces to be printed. For example: 30 PRINT "HELLO";SPC(5);"TOM" HAS OUTPUT: HELLO TOM (5 SPACES BETWEEEN WORDS) The SPC function is not available on all systems.	Have students print his name across the screen, with 4 blank spaces between each occurrence of the name.

Name_____

Period_____

SAMPLE TEST

TEST ON TOPIC SIX

1. What will be printed as a result of executing this program?
(5 pts.)

```
10 LET S = 0
20 FOR K = 7 TO 13 STEP 2
30 LET S = S + K
40 NEXT K
50 PRINT S
60 END
```

2. What will be printed as a result of executing this program?
(5 pts.)

```
10 LET A = 0
20 FOR X = 1 TO 5
30 READ B
40 LET A = A + B
50 NEXT X
60 LET S = A/5
70 PRINT S
80 DATA 80, 85, 90, 95, 100
90 END
```

3. Label the spaces and give the exact printout. Circle the system you are using:
MPM TRS-80
(10 pts.)

```
10 PRINT "NUMBER", "SQUARE"
20 FOR I = ITO 5
30 PRINT I, I + 2
40 NEXT I
50 END
```

4. Label the spaces and give the exact printout.
(10 pts.)

```
10 READ A, B
30 PRINT A; B; A + B
35 GO TO 10
40 DATA 2, 3, 4, -6, 10, 12
50 END
```

5. Write a print statement using the TAB function that will produce the following output.
(5 pts.)

				F	I	R	S	T						S	E	C	O	N	D

Which of the following are valid PRINT statements?
(4 pts. each)

Yes or No:

_____ 6. 80 PRINT "AREA" = A

_____ 7. 20 PRINT X; Y; Z

_____ 8. 15 PRINT

_____ 9. 20 PRINT "SIDE =" S "PERIM =" 4 * 5

_____ 10. "(" X ", Y ") "

Which of the following are valid BASIC statements?

_____ 11. LET I - 1 = I

_____ 12. LET K2 = K2 + 2

_____ 13. FOR J = 1, 10

_____ 14. NEXT I + 1

15. In entering statement 30, K = 5, what value is assigned to K by this statement? (4 pts.)

_____ 30 K = K + 1

16. Find the error in this program and circle it.

```
10 FOR I = 1 TO 5
20 S = 0
30 READ A
40 S = S + A
50 NEXT I
60 PRINT S
70 DATA 2, 7, 12, 3, 9
80 END
```

17. Write a flowchart and program to sum the even numbers from 2 to 24. (20 pts.)

Bonus (10 pts.) Give #'s of all spaces.)

Exactly what will be printed when this program is run?

```
10 LET I = 1
20 PRINT TAB (2 + I + 1)
30 LET I = I + 1
40 IF I < = 4 THEN 20
50 PRINT "THAT'S ENOUGH"
60 END
```

ANSWER KEY

SAMPLE TEST

TEST ON TOPIC SIX

1. What will be printed as a result of executing this program? (5 pts.) 40

```
10 LET S = 0
20 FOR K = 7 TO 13 STEP 2
30 LET S = S + K
40 NEXT K
50 PRINT S
60 END
```

2. What will be printed as a result of executing this program? (5 pts.) 90

```
10 LET A = 0
20 FOR X = 1 TO 5
30 READ B
40 LET A = A + B
50 NEXT X
60 LET S = A/5
70 PRINT S
80 DATA 80, 85, 90, 95, 100
90 END
```

3. Label the spaces and give the exact printout. Circle the system you are using:
MPM TRS-80
(10 pts.)

```
10 PRINT "NUMBER", "SQUARE"
20 FOR I = ITO 5
30 PRINT I, I↑2
40 NEXT I
50 END
```

1	2	3	4	5	6	7
N	U	M	B	E	R	
1						
2						
3						
4						
5						

17	18

TRS-80

16	17	18	19	20	21
	1				
	4				
	9				
	16				
	25				

MPM

4. Label the spaces and give the exact printout.
(10 pts.)

```
10 READ A, B
30 PRINT A; B; A + B
35 GO TO 10
40 DATA 2, 3, 4, -6, 10, 12
50 END
```

1	2	3	4	5	6	7	8	9	10	11
	2			3			5			
	4		-	6		-	2			
	1	0			1	2			2	2

5. Write a print statement using the TAB function that will produce the following output.
(5 pts.)

				F	I	R	S	T						S	E	C	O	N	D

CHECK YOUR MACHINE)

```
PRINT TAB (4) "FIRST";  TAB (13) "SECOND"
    or
PRINT TAB (5) "FIRST";  TAB (14) "SECOND"
```

Which of the following are valid PRINT statements?
(4 pts. each)

Yes or No:

No 6. 80 PRINT "AREA" = A

Yes 7. 20 PRINT X; Y; Z

Yes 8. 15 PRINT

Yes 9. 20 PRINT "SIDE =" S "PERIM =" 4 * 5

No 10. "(" X ", Y ") "

Which of the following are valid BASIC statements?

No 11. LET I - 1 = I

Yes 12. LET K2 = K2 + 2

No 13. FOR J = 1, 10

No 14. NEXT I + 1

15. In entering statement 30, K = 5, what value is assigned to K by this statement?
(4 pts.)

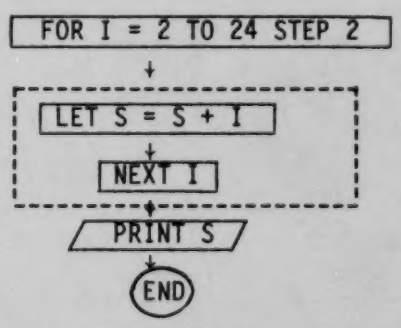
6 30 K = K + 1

16. Find the error in this program and circle it.

```
10 FOR I = 1 TO 5
20 S = 0
30 READ A
40 S = S + A
50 NEXT I
60 PRINT S
70 DATA 2, 7, 12, 3, 9
80 END
```

17. Write a flowchart and program to sum the even numbers from 2 to 24.
(20 pts.)

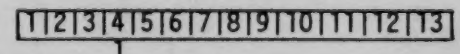
```
10 FOR I=2 TO 24 STEP 2
20 S = S + I
30 NEXT I
40 PRINT S
50 END
```



Bonus (10 pts.) Give #'s of all spaces.)

Exactly what will be printed when this program is run?

```
10 LET I = 1
20 PRINT TAB (2 + I + 1)
30 LET I = I + 1
40 IF I < = 4 THEN 20
50 PRINT "THAT'S ENOUGH"
60 END
```



2
3
4
THAT'S ENOUGH
↑ 13th

128

TOPIC SEVEN: BASIC FUNCTIONS

Performance Objectives:

The student will -

A13 - Write BASIC programs using available library functions including RND, INT, ABS, SGN, SQR.

A13a - Write BASIC progams using available library functions including SIN, COS, TAN, ATN, LOG, EXP.

A17(i) - Produce n distinct random integers for $n \leq 50$.

A17(j) - Accept three integers m, n, and k, and randomly generate k integers between m and n.

A17(k) - Manipulate numerical data and round the results to a given number of decimal places.

A17(l) - Use coordinate geometry formulas to make decisions about geometric figures.

References:

Textbook, pp. 203-225

COMPUTER MATHEMATICS

Objectives A17(j): The student will write BASIC programs that will accept three integers m, n, and k, and randomly generate k integers between m and n.

A17(k): The student will manipulate numerical data and round the results to a given number of decimal places.

A17(l): The student will use coordinate geometry formulas to make decisions about geometric figures.

Textbook Reference: Pp. 209-213

TEACHING SUGGESTIONS

Remember that on most systems RND(Ø) (or RND(1)) never reaches the value 1 exactly. Therefore, INT (5Ø*RND(Ø)) takes on integral values from 0 to 49.
Thus

```
LET X = M + INT ((N-M+1)*RND(Ø))
```

or

```
LET X = INT ((N-M+1)*RND(Ø)+M)
```

chooses a random number (integer) between M and N, inclusive.

Students frequently need considerable practice to write and interpret such a statement correctly. Give several examples such as

a) integers between 3 and 17, inclusive
b) integers between -10 and 10, inclusive and even
c) real numbers between 3 and 17

Answers:

a) LET X = 3 + INT (15*RND(Ø))
b) LET X = -10 + INT (21*RND(Ø))
c) LET X = 3 + 14 * RND(Ø)

Note the 15 in answer (a) and the 14 in answer (c).

Numbers are easily rounded with the help of the INT function. Thus 0.1*INT(10*X+.5) rounds X to the nearest tenth and 10*INT(0.1*X+.5) rounds X to the nearest ten. See Program 17 (k) of the Program Appendix for more examples.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral or Written
Problems 4, 5 on p. 213

Note: The book's answer to problem 4 is incorrect. It should be

70 LET X = INT (14 * RND(Ø))

Supplement A17(l) to satisfy objective A17(l), an application of ABS and SQR functions.

COMPUTER MATHEMATICS

OBJECTIVE A13: The student will write BASIC programs using available library functions including KND(X), INT(X), ABS(X), SGN(X), and SQR(X).

A13a: The student will use library functions including SIN(X), COS(X), TAN(X), and EXP(X).

A17(i): The student will write BASIC program to produce n distinct random integers for n<50.

Textbook Reference: Pp. 203-225

TEACHING SUGGESTIONS

Check system used for available functions. Sample list on page 218 of textbook.

When using RND function, check system to determine effects of different arguments.

For example:

a) Which is used to generate a list of random (actually pseudo-random) members, RND(0) or RND(1)?
b) Does RND(2) differ in range from RND(1)?
c) Does RND(-1) give the same list of numbers as RND(0) or RND(1)?

Figure 7.9, on page 211 of textbook is a program for Objective 17(i).

Errors in the Textbook

The book's answer to 4C on p. 219 is also incorrect. It should be:

C. Range between 0 and 98 inclusive integers only.

#1, p. 213 says to repeat executing the program until all the data is used. No matter how many times the program is run, only the first X value, 1, will be used.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral
- Exercise One, pp. 206-207, #1-4
- Exercise Two, p. 213, #1-3
- Exercise Threee (optional) p. 217, #1-3
- Objective Activity, pp. 219-220, #1-9

Written
- Problem Set Seven, pp. 223-224, #1-9

Programs
- Problem Set Seven, pp. 224-225, #10-14, #15-18 (optional)

Change the program to read:

```
80 READ X
90 DATA 1,2,37,55,24,88,13,64
100 IF INT(X/2)*2 = X THEN 130
120 GOTO 80
130 PRINT X, "YES"
140 GOTO 80
```

```
1          NO
2          YES
37         NO
55         NO
24         YES
88         YES
13         NO
64         YES
```

Errors in Teacher's Guide

In Problem Set Seven, the Teacher's Guide has several incorrect answers:

P. 60

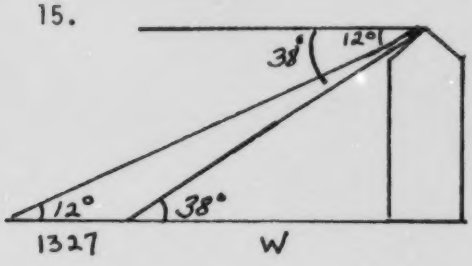
5e. should be -11
6a. should be 40 IF 7/11 = INT (7/11) THEN 70
d. should be 40 IF NY = INT (X/Y) THEN 70
7a. should generate integers 1-13
20 LET X = INT (RND(Ø) * 13 + 1)
b. 20 LET X = INT (RND(Ø) * 50 + 1)
c. should generate integers 2-24 since word inclusive is not used
20 LET X = INT (RND(Ø) * 23 + 2)

14. This program is not appropriate because it uses subscripted variables which are not used until the next chapter.

The program for #14 is actually given in the textbook as Fig. 7.11, p. 212.

Problems #15 and #16 on p. 224 involve knowledge of trigonometry, and not all students will be able to do the math calculations.

15.



Tower Tan 38° = $\frac{T}{W}$, Tan 12° = $\frac{T}{1327+W}$

Let X = Tan 38°, Y = Tan 12°

$\therefore X = \frac{T}{W}$, $Y = \frac{T}{1327+W}$

$W = \frac{T}{X}$, substituting we have $Y = \frac{T}{1327 + \frac{T}{X}}$

Solving for T: $T = 1327 \cdot Y + Y\left(\frac{T}{X}\right)$

$$T - \frac{YT}{X} = 1327 \cdot Y$$

$$\frac{T(1 - Y)}{X} = 1327 \cdot Y$$

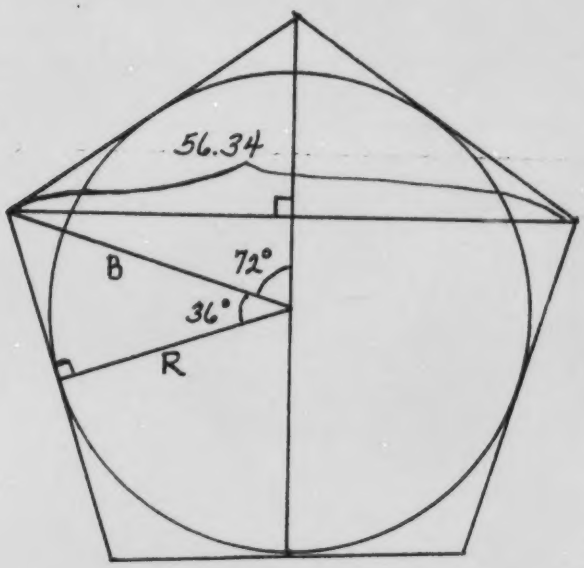
$$T = \frac{1327 \cdot Y}{1 - Y/X}$$

$$= \frac{1327 \cdot Y}{\frac{X - Y}{X}}$$

$$= 1327Y \cdot \frac{X}{X-Y}$$

$$= \frac{1327 \cdot Y \cdot X}{X-Y}$$

16.



$\sin 72° = (X/2)/B$

$\therefore B = (X/2)/ \sin 72°$

$\cos 36° = R/B$

$\therefore R = B \cdot \cos 36°$

COORDINATE GEOMETRY

1. If given the coordinates of 3 noncollinear points, decide whether the triangle formed by joining them is isosceles, equilateral, or scalene.

```
 10 READ X1, Y1, X2, Y2, X3, Y3
 15 IF X1 = 99 THEN END
 20 D1 = SQR ((X2-X1) ↑2 + (Y2-Y1) ↑2)
 30 D2 = SQR ((X3-X2) ↑2 + (Y3-Y2) ↑2)
 40 D3 = SQR ((X1-X3) ↑2 + (Y1-Y3) ↑2)
 50 IF ABS (D1-D2) < .01 THEN 100
 60 IF ABS (D1-D3) < .01 THEN 110
 70 IF ABS (D2-D3) < .01 THEN 110
 80 PRINT "TRIANGLE IS SCALENE"
 90 GO TO 10
100 IF ABS (D2-D3) < .01 THEN 130
110 PRINT "TRIANGLE IS ISOSCELES"
120 GO TO 10
130 PRINT "TRIANGLE IS EQUILATERAL"
140 GO TO 10
150 DATA 4, 2, 7, -1, 7, 5, 0, 0, 2, 4, 4, 4, 0, 0, 1, 1.732, 2, 0
165 DATA 99, 99, 99, 99, 99, 99
170 END
```

Note: Use of tolerances in lines 50, 60, 70 and 100 of this program, necessary because of the possibility of round-off error in the date (1.732) and in raising a number to the second power.

2. Given the coordinates of 2 points in the xy-plane, print the equation of the line through the 2 points.

```
 10 READ X1, Y1, X2, Y2
 15 IF X1 = 99 THEN 100
 20 IF X2 - X1 = 0 THEN 70
 30 M = (Y2-Y1) / (X2-X1)
 40 C = M * X1 - Y1
 50 PRINT M "X - Y = " C
 60 GO TO 10                              (Printout --> 2X-Y=5)
 70 PRINT "X - Y = " C                                 X = 3
 80 GO TO 10
 90 DATA 6, 7, 8, 11, 3, 2, 3, -1, 99, 99, 99, 99
100 END
```

3. Given the coordinates of 4 points in the xy-plane, determine whether the quadrilateral formed by joining the points in order is equilateral. Assume points are input in clockwise or counterclockwise order.

```
 10 READ X1, Y1, X2, Y2, X3, X4, Y4
 20 D1 = SQR ((X2-X1)↑2 + (Y2-Y1)↑2)
 30 D2 = SQR ((X3-X1)↑2 + (Y3-Y2)↑2)
 40 IF ABS (D1-D2) > .01 THEN 100
 50 D3 = SQR ((X4-X3)↑2 + (Y4-Y3)↑2)
 60 IF ABS (D3-D2) > .01 THEN 100
 70 D4 = SQR ((X1-X4)↑2 + (Y1-Y4)↑2)
 80 IF ABS (D4-D1) > .01 THEN 100
 90 PRINT "THE QUADRILATERAL IS EQUILATERAL"
 95 GO TO 10
100 PRINT "THE QUADRILATERAL IS NOT EQUILATERAL"
105 GO TO 10
110 Data -3,4,2,4,2,-1,-3,-1,0,1,4,3,7,-1,2,-3,-3,2,-1,0,-3,-2,-5,0
120 END
```

```
Printout --- THE QUADRILATERAL IS EQUILATERAL
             THE QUADRILATERAL IS NOT EQUILATERAL
             THE QUADRILATERAL IS EQUILATERAL
```

Name _____

Period_____

SAMPLE TEST

TOPIC SEVEN

(5 pts each)
Correct these BASIC function expressions:

1. 40 LET X = INT (SQR(9)

2. 20 LET X = RDN (0) _____

3. 35 LET X = INT 6.3

4. 70 LET X = INT (ABS(3)-1

Find the value of X in each of these expressions.

5. 10 LET X = SQR (INT(9.3)) _____

6. 10 LET X = ABS (INT(-3.4) _____

7. 10 LET X = ABS (13-28) _____

8. 10 LET X = SQR(SQR(81)) _____

Determine the range of the random numbers that would be generated by the computer for each of the following LET statements. Indicate which statements only generate integers.

9. 40 LET X = RND (0) _____

10. 40 LET X = 15 * RND(0) _____

11. 40 LET X = INT(RND(0)*10+1) _____

12. 40 LET X = INT(RND(0)*100) _____

13. Write an IF-THEN statement of the form 40 IF-THEN 70 to test if A is a factor of B.

_____

14. Write a BASIC statement to generate numbers between 0 and 20.

_____

15. Write a BASIC statement to generate integers between 1 and 50 inclusive.

_____

Write a valid LET statement for the following algebraic expressions.

16. $X = \sqrt{1a^2 + 2a1}$ _____

17. $X = \sqrt{3.1416}$ _____

18. Evaluate A = SQR(ABS(X-Y)) for X = 58 and Y = 94. _____

19. Evaluate A = INT(Y/X) if X = 12 and Y = 11. _____

20. Evaluate A = ABS (Y-X) if X = 12 and Y = 11. _____

Bonus: (8 pts.) Write a BASIC statement using the RND and INT functions to generate integers between 12 and 60 inclusive.

ANSWER KEY

SAMPLE TEST

TOPIC SEVEN

(5 pts each)
Correct these BASIC function expressions:

1. 40 LET X = INT (SQR(9))
2. 20 LET X = RDN (0) RND
3. 35 LET X = INT (6.3)
4. 70 LET X = INT (ABS(3)-1)

Find the value of X in each of these expressions.

5. 10 LET X = SQR (INT(9.3)) 3
6. 10 LET X = ABS (INT(-3.4) 4
7. 10 LET X = ABS (13-28) 15
8. 10 LET X = SQR(SQR(81)) 3

Determine the range of the random numbers that would be generated by the computer for each of the following LET statements. Indicate which statements only generate integers.

9. 40 LET X = RND (0) 0 < X < 1
10. 40 LET X = 15 * RND(0) 0 < X < 15
11. 40 LET X = INT(RND(0)*10+1) 1 < X < 10 integers
12. 40 LET X = INT(RND(0)*100) 0 < X < 99 integers
13. Write an IF-THEN statement of the form 40 IF-THEN 70 to test if A is a factor of B.

    40 IF B/A = INT (B/A) THEN 70
14. Write a BASIC statement to generate numbers between 0 and 20.

    LET X = RND (Ø) * 20
15. Write a BASIC statement to generate integers between 1 and 50 inclusive.

    LET X = INT (RND(Ø) * 50 + 1)

Write a valid LET statement for the following algebraic expressions.

16. $X = \sqrt{1a^2 + 2a1}$ X = SQR(ABS(A↑2 + 2 * A))

17. $X = \sqrt{3.1416}$ X = SQR (3.1416)

18. Evaluate A = SQR(ABS(X-Y)) for X = 58 and Y = 94. 6

19. Evaluate A = INT(Y/X) if X = 12 and Y = 11. 0

20. Evaluate A = ABS (Y-X) if X = 12 and Y = 11. 1

Bonus: (8 pts.) Write a BASIC statement using the RND and INT functions to generate integers between 12 and 60 inclusive.

LET X = INT (RND(Ø) * 49 + 12)

TOPIC EIGHT: SUBSCRIPTING

Performance Objectives:

The student will -

A14 - Write BASIC programs which utilize single subscripts.

A17(b) - Calculate n! for n = 1 to 15.

A17(f) - Accept any integer with a maximum of 6 digits and print out the value of each digit and the value of the position it occupies.

A17(g) - Generate n terms of the Fibonacci sequence for n < 25.

References:

Textbook, pp. 227-259

Objective A14: The student will write BASIC programs which utilize single subscripts.

A17(f) The student will write a BASIC program that will accept any integer with a maximum of 6 digits print out the value of each digit and the value or the position it occupies.

Textbook Reference: Pp. 227-259

TEACHING SUGGESTIONS

Explain purposes of subscripted variables

1) to increase number of available variables
2) to simplify selection of individual numbers from a list

Be sure students understand difference between simple variables that include letter and digit A3, X9 and subscripted variable A(3), X(9).

Subscripts may be numerals, variables, or expressions. Examples of correct subscripted variables:

```
A(3)
B(X)
C(Y-3)
G(M(INT(X+3*(RND(1)))))
```

Remind students that a DIM statement can be "passed through" only once.

Objective A17(f) provides a good problem for the use of subscripts. See Programs A17(f) in the Program Appendix.

Check your system to determine when DIM statements are required. On some systems DIM is required only when the number of elements in list exceeds 10. It is, however, good practice to use DIM whenever subscripts are used. Also check whether or not subscript Ø is allowed.

Remember: DIM statements indicate the maximum number of elements allowed in list. Be sure students do not confuse assigned value of variable with value of subscript.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral
Exercise One - Pp. 230-231, #1-3
Objective Activity, p. 249, #1-3

Written
Problem Set Eight, p. 254, #1-4

Textbook

Oral
Exercise Two, pp. 233-234, #1-3
Objective Activity, p. 249, #4-5

Written
Problem Set, p. 255, #6-7

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Review FOR-NEXT loops. Then do example(s) (pages 236 and 238 in textbook) of use of FOR-NEXT to designate subscript for INPUT and OUTPUT. Use of trace is invaluable in checking for errors in programs involving FOR-NEXT and subscripts. Corrections to Teacher's Guide Answers to Problem Set 8, p. 64. p. 64: 10e. Answer should be 7 to PRINT X(2) + X(3) (my book has PTINT) 11a. Six numbers are printed on the first line - I don't know of any computers with 6 zones. Printout would be - 1 2 3 4 5 or 1 2 3 4 5 4 3 2 1 5 5 4 3 2 1	Oral Exercise Three, p. 240, #1-5 Objective Activity, p. 250, #6-7 Written Problem Set, pp. 255-258, #5, 8-14

Objective A14: The student will write BASIC programs which utilize single subscripts.

A17(b): The student will calculate n! for n = 1 to 15.

A17(g): The student will generate n terms of the Fibonacci series for n<25 and find their sum.

Textbook Reference: Pages 243-259

TEACHING SUGGESTIONS

Go over a number of examples with students at board.

Be sure they can do a trace for programs including subscripts and loops.

Have students working together at board work through one problem from beginning to end - variables, flowchart, program, trace.

At this point the logic involved becomes even more important.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral
Exercise Four, pages 246-247, #1-3

Objective Activity - pages
Pages 250-251, #8-10

Problem 9:

Fig. 87.27 mp. 253 solves Objective A17(b) except for column headings, if line 110 is changed to DATA 15. See also Program A17(b) in Program Appendix.

Problem 10:

Fig. 8.29 on p. 255 solves Objective A17(g) for n=20, except athat the sum is not found. See also Supplement 17(g). 1.

#16. A better program utilizing the previous chapter might be:

```
10 DIM A(20)
20 READ A(1), A(2)
30 FOR I = 3 TO 20
40 IF I/2 = INT (I/2) THEN TO
50 LET A(I) = A(1)
60 GO TO 80
70 LET A(I) = A(2)
80 NEXT I
90 DATA 2.5, 7.1
100 END
```

Supplement A17(g): - Extended Fibonacci Sequences

The Fibonacci sequence 1, 1, 2, 3, 5, 8, 13, ... had the property that each number, after the second, is the sum of the previous two.

The sequence 1, 1, 1, 3, 5, 9, 17, ... has the property that each number, after the third, is the sum of the previous three.

Write a program which will print the first n terms of this sequence.

Solution:

```
10 REM EXTENDED FIBONACCI SEQUENCE
20 INPUT "NUMBER OF TERMS?"; N
30 DIM F (N)
40 F(1) = 1: F(2) = 1: F(?) = 1
50 FOR I = 1 < THEN 70
60 F (I) = F(I-1) + F(I-2) + F(I-3)
70 PRINT I, F(I)
80 NEXT I
90 END
```

As a further extension, for K an integer > 2, consider the sequence whose first K terms are 1, and where each subsequent term is the sum of the previous K.

Write a program which will print the first n terms of this K-sequence. As an added challenge, write the program so that the extended Fibonacci numbers F(I) are calculted subject to the dimension statement DIMF(K), no matter how many terms are to be calculated.

For a solution, see program A17(g) in the Program Appendix.

Computer Math

Name_____
Period_____

S A M P L E

Test on Topic 8

True-False 25 Pts.

_____ 1. DIM stands for dimension.

_____ 2. DIM A(50) means that A(I) cannot store values that are larger than 50.

_____ 3. If B = [38 24 97 88 102] then the DIM statement needed for the array is B[102].

_____ 4. A DIM statement can never be in a loop.

_____ 5. The number of values in a list may be less than the value in the DIM statement but never more.

Let the array be [8 -4 16 -12 2 11]
with the indexing beginning at one.
Give the value of each of the following. 25 Pts.

_____ 6. A(2)

_____ 7. A(3)-1

_____ 8. 2*A(4)

_____ 9. A(A(5))

_____ 10. A(3-1)

What will be printed out as a result of executing this program? 5 Pts.

_____ 11.
```
 10 DIM A(12)
 20 FOR I = 1 TO 12
 30 READ A(I)
 40 NEXT I
 50 S = 0
 60 FOR I = 1 TO 12 STEP 3
 70 S = S + A (I)
 80 NEXT I
 90 PRINT S
100 DATA 6, 4, 3, 5, 9, 2, 15, 21, 10, 7, 1, 13
110 END
```

12. Show the output of the following program. 10 Pts.

_____

```
10 DIM A (10)
20 A(1) = 1
30 A(2) = 1
40 FOR I = 3 TO 10
50 LET A(I) = A(I-1) + A(I-2)
60 NEXT I
70 FOR I = 1 TO 10
80 PRINT A(I);
90 NEXT I
100 END
```

13. Show the output. 10 Pts.

```
10 DIM A(10)
20 READ N
30 LET A(1) = 1
40 FOR I = 2 TO N
50 LET A(2) = I* A(I-1)
60 NEXT I
70 FOR I = 1 TO N
80 PRINT A (I)
90 NEXT I
100 DATA 5
110 END
```

14. Construct a valid DIM statement for the following DATA statement using the variable indicated. 15 Pts.

95 DATA 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 (Q-list)

_____

15. Write a program to read 20 #'s and print out every # starting with the first one read. 20 Pts.

Computer Math

ANSWER KEY

Name_____
Period_____

SAMPLE

Test on Topic 8

True-False 25 Pts.

True 1. DIM stands for dimension.

False 2. DIM A(50) means that A(I) cannot store values that are larger than 50.

False 3. If B = [38 24 97 88 102] then the DIM statement needed for the array is B[102].

True 4. A DIM statement can never be in a loop.

True 5. The number of values in a list may be less than the value in the DIM statement but never more.

Let the array be [8 -4 16 -12 2 11]
with the indexing beginning at one.
Give the value of each of the following. 25 Pts.

-4 6. A(2)

15 7. A(3)-1

-24 8. 2*A(4)

-4 9. A(A(5))

-5 10. A(3-1)

What will be printed out as a result of executing this program? 5 Pts.

33 11.
```
10 DIM A(12)
20 FOR I = 1 TO 12
30 READ A(I)
40 NEXT I
50 S = 0
60 FOR I = 1 TO 12 STEP 3
70 S = S + A (I)
80 NEXT I
90 PRINT S
100 DATA 6, 4, 3, 5, 9, 2, 15, 21, 10, 7, 1, 13
110 END
```

12. Show the output of the following program. 10 Pts.

1 1 2 3 5 8 13 21 34 55

```
10 DIM A (10)
20 A(1) = 1
30 A(2) = 1
40 FOR I = 3 TO 10
50 LET A(I) = A(I-1) + A(I-2)
60 NEXT I
70 FOR I = 1 TO 10
80 PRINT A(I);
90 NEXT I
100 END
```

13. Show the output. 10 Pts. 1 2 6 24 120

```
10 DIM A(10)
20 READ N
30 LET A(1) = 1
40 FOR I = 2 TO N
50 LET A(2) = I* A(I-1)
60 NEXT I
70 FOR I = 1 TO N
80 PRINT A (I)
90 NEXT I
100 DATA 5
110 END
```

14. Construct a valid DIM statement for the following DATA statement using the variable indicated. 15 Pts.

```
95 DATA 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12      (Q-list)
```

DIM Q(12)

15. Write a program to read 20 #'s and print out every # starting with the first one read. 20 Pts.

```
10 FOR I = 1 TO 20
20 READ A(I)
30 NEXT I
40 DATA
50 FOR I = 1 TO 20 STEP 2
60 PRINT A(I);
70 NEXT I
```

TOPIC NINE: NESTED LOOPS AND SORTING

Performance Objectives:

The student will -

A12 - Flowchart and write BASIC programs utilizing nested FOR-NEXT loops.

A17(E) - Select and print out the largest and smallest numbers in a set of data.

References:

Textbook - pages 261-293

Objective A12: The student will flowchart and write programs utilizing nested FOR-NEXT loops.

Textbook Reference: Pages 261-265

TEACHING SUGGESTIONS

Trace and compare the programs on page 261 and page 263 to show students importance and power of nested FOR-NEXT loops.

Note:

When constructing nested loops take note -

1) Each loop must have a different index

2) The inner loop must be completely nested within the outer loop.

WRONG	RIGHT
10 FOR X	10 FOR X
20 FOR Y	20 FOR Y
20 NEXT X	30 NEXT Y
40 NEXT Y	40 NEXT X

3) A loop may be exited (inner or outer) at any step, but a loop may be entered ONLY at the FOR statement.

   An exception to this rule would be an exit to a subroutine and reentry inside the loop.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral

Exercise One - pages 264-462 #1-3

Objective Activity - page 283 #1-3

Written

Problem Set Nine, page 292 #1-2

Supplement 12.6 - Parkside's Triangle

Objective A12 - continued

A17(e) - The student will select and print out the largest and smallest numbers in a set of data.

Textbook Reference: Pages 267 - 271

TEACHING SUGGESTIONS

Selecting largest or smallest number from a list

First go over the simple non-subscripted flowchart or program on page 267 of textbook for the logic of selection.

Then trace with different INPUT's the subscripted flowchart a program on pp. 268 or 269.

Stress the flexibility of this second type flowchart or program.

The program on p. 269 selects the largest member in a set or data. The program on p. 271 selects the smallest. A combination of the two will therefore give a satisfactory program for Objective A17(e).

The program on p. 271 may be simplified for students who have difficulty with the concept to:

```
10 REM SMALLEST NUMBER TO AN ARRAY
15 READ N
20 FOR K = 1 TO N
25 READ R(K)
30 NEXT K
35 LET S = R(1)
40 FOR I = 2 TO N
45 IF S < R(1) THEN 55
50 LET S = R(1)
55 NEXT I
60 PRINT "THE SMALLEST NUMBER OF HOURS IS"; S
65 DATA 7,45.5,43,42,25,44.5
70 DATA 41,43.5,42
75 END
```

THE SMALLEST NUMBER OF HOURS IS 41.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook:

Oral
Exercise Two, pages 270-271 #1-4

Objective Activity
Pages 283 #4-6
On p. 283 - Objective Activity #4, line 80 should read GO TO 100, not 110. The answer on p. 286 has the same error.

Written
Problem Set Nine, p. 292 #3

Objective A12 - continued

A17(m) - Arrange a set of numbers in order from least to greatest.

Textbook Reference: Pp. 272-293

TEACHING SUGGESTIONS

SORTING

First go over SWITCHING. Then combine that and the selection of largest (or smallest) number program.

Do traces.

Most students should not be expected to prepare a SORTING program under test conditions.

However as an assignment all students should with use of book, do these programs.

Fig. 9.27 on p. 282 of textbook illustration Objective A17(m).

There are many different sorting algorithms. To help students understand these methods, arrays of numbers can be used. The array is rewritten each time a change is made. Having arrayed different arrays of numbers using the algorithm, the students can then grasp the computer program more readily.

See the following:

The array from p. 291 (Fig. 9.37) hand-run and illustrated on the computer.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral

Exercise Three, pp. 274-275 #1-3

Exercise Four, p. 281 #1-3

Objective Activity, pp. 284-285 #7-14

Written

Problem Set Nine, pp. 293 #7-10

Corrections:

On page 69-71 of the Teacher's Guide, #9 and #10 from p. 293 in the text are incorrect. The programs sort from smallest to largest--but the book says to use #8, p. 293, which arranges a list from largest to smallest

Programs

Problem Set Nine, p. 293 #11

On page 279 in Topic 9, the output of the sort following the flowchart in Figure 9.20 is given as

68
42
7

This is incorrect. The flowchart sorts from smallest to largest. A reversal of the YES and NO in the inner loop would make the flowchart sort from largest to smallest. The corresponding program on p. 282 does sort from largest to smallest.

Supplement A12.6 - Parkside's Triangle

Bonus - a followup to #2, p. 292 - Problem Set Nine of Textbook

Like problem #2, p. 292, Parkside's Triangle has the same number of elements on each row as that row's number, but the elements do not have to all be the same as the row number. Instead, the numbers range from 1 to 9 (consecutively). The initial number need not be 1, however. Therefore, there are two numbers that are important to any of Parkside's Triangles. The row size number (R) indicates how many rows and the seed number or initializer (S) indicates the first element of the triangle.

Here are some examples of Parkside's Triangle:

Row Size 5, Seed 1

```
1
2 3
4 5 6
7 8 9 1
2 3 4 5 6
```

Row Size 7, Seed 3

```
3
4 5
6 7 8
9 1 2 3
4 5 6 7 8
9 1 2 3 4 5
6 7 8 9 1 2 3
```

The problem is to write a program which will generate any Parkside's Triangle with no more than 15 rows ($R \leq 15$) and any seed from 1 to 9 ($1 \leq X \leq 9$). Use the following data to test your program. R=6, S=1 and R=8, S=5.

Sample program:

```
 10 REM PARKSIDE'S TRIANGLE
 20 READ R,S
 30 IF R>15 OR S<1 OR S>9 THEN END
 40 DATA 5, 1, 8, 5, 99, 99
 50 K = 0
 60 P = S - 1
 70 FOR I = 1 TO R
 80 P = P + 1
 90 K = K + 1
100 IF P>9 THEN P = 1
110 PRINT P;
120 IF K<I THEN 80
130 K = 0
140 PRINT
150 NEXT I
160 PRINT
170 GO TO 20
180 END
```

The sort algorithm for the program on p. 291 (Fig. 9.37).

(The array is rewritten each time a change is made.

The smallest number is found and placed in A(1). The number originally stored as A(1) is then placed in the former location of the smallest number. This process is repeated, placing one number in the correct order after each switch.

A(1)	A(2)	A(3)	A(4)	A(5)	A(6)	A(7)	A(8)	A(9)	A(10)
27.7	13.2	40.1	-56.4	26.7	13.2	44.0	12.4	-26.7	100.6
-56.4	13.2	40.1	27.7	26.7	13.2	44.0	12.4	-26.7	100.6
-56.4	-26.7	40.1	27.7	26.7	13.2	44.0	12.4	13.2	100.6
-56.4	-26.7	12.4	27.7	26.7	13.2	44.0	40.1	13.2	100.6
-56.4	-26.7	12.4	13.2	26.7	27.7	44.0	40.1	13.2	100.6
-56.4	-26.7	12.4	13.2	13.2	27.7	44.0	40.1	26.7	100.6
-56.4	-26.7	12.4	13.2	13.2	26.7	44.0	40.1	27.7	100.6
-56.4	-26.7	12.4	13.2	13.2	26.7	27.7	40.1	44.0	100.6

When A(8) is compared to A(9) and A(10) no changes are made.
When A(9) is compared to A(10), no change is made.

```
10 DIM A(100)
20 READ N
30 PRINT "THE ORIGINAL ARRAY"
40 FOR M = 1 TO N
50 READ A(M)
60 PRINT A(M);
70 NEXT M
80 PRINT
90 PRINT "THE EXCHANGES IN THE ARRAY ARE:"
100 DATA 10,27.7,13.2,40.1,-56.4,26.7,13.2,44.0,12.4,-26.7,100.6
110 FOR K = 2 TO N
120 LET L = K-1
130 FOR I = K TO N
140 IF A(L) <=  A(I) THEN 160
150 LET L = I
160 NEXT I
170 LET T = A(K-1)
180 LET A(K-1)=A(L)
190 LET A(L) = T
200 FOR M = 1 TO N
210 PRINT A(M);
220 NEXT M
230 PRINT
240 NEXT K
250 END
```

```
THE ORIGINAL ARRAY
 27.7  13.2  40.1 -56.4  26.7  13.2  44  12.4 -26.7  100.6
THE EXCHANGES IN THE ARRAY ARE:
-56.4  13.2  40.1  27.7  26.7  13.2  44  12.4 -26.7  100.6
-56.4 -26.7  40.1  27.7  26.7  13.2  44  12.4  13.2  100.6
-56.4 -26.7  12.4  27.7  26.7  13.2  44  40.1  13.2  100.6
-56.4 -26.7  12.4  13.2  26.7  27.7  44  40.1  13.2  100.6
-56.4 -26.7  12.4  13.2  13.2  27.7  44  40.1  26.7  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  44  40.1  27.7  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  40.1  44  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  40.1  44  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  40.1  44  100.6
```

A different sequential sorting program follows. It may be easier for some students to understand. The array is printed after each interchange.

```
10 REM A SEQUENTIAL SORT
20 READ N
30 DIM A(100)
40 PRINT "THE ORIGINAL ARRAY"
50 FOR M = 1 TO N
60 READ A(M)
70 PRINT A(M);
80 NEXT M
90 PRINT
100 PRINT"THE EXCHANGES IN THE ARRAY ARE :"
110 DATA 10,27.7,13.2,40.1,-56.4,26.7,13.2,44.1,12.4,-26.7,100.6
120 FOR I = 1 TO N-1
130 FOR J = I TO N
140 IF A(I) <= A(J) THEN 180
150 LET T = A(I)
160 LET A(I)=A(J)
170 LET A(J)=T
180 NEXT J
190 FOR K = 1 TO N:PRINT A(K);:NEXT K:PRINT
200 NEXT I
210 END
```

```
THE ORIGINAL ARRAY
 27.7  13.2  40.1 -56.4  26.7  13.2  44.1  12.4 -26.7  100.6
THE EXCHANGES IN THE ARRAY ARE :
-56.4  27.7  40.1  13.2  26.7  13.2  44.1  12.4 -26.7  100.6
-56.4 -26.7  40.1  27.7  26.7  13.2  44.1  13.2  12.4  100.6
-56.4 -26.7  12.4  40.1  27.7  26.7  44.1  13.2  13.2  100.6
-56.4 -26.7  12.4  13.2  40.1  27.7  44.1  26.7  13.2  100.6
-56.4 -26.7  12.4  13.2  13.2  40.1  44.1  27.7  26.7  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  44.1  40.1  27.7  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  44.1  40.1  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  40.1  44.1  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  40.1  44.1  100.6
```

All of these sorting programs are slow. For more sophisticated sorting methods see Supplement B17a(j).

The following program is a simple bubble sort. Adjacent positions in the array are compared, and numbers seem to "bubble" to the front or back of the array.

```
10 REM A BASIC SORT
20 READ N
30 DIM A(100)
40 PRINT "THE ORIGINAL ARRAY"
50 FOR M = 1 TO N
60 READ A(M)
70 PRINT A(M);
80 NEXT M
90 PRINT
100 PRINT"THE EXCHANGES IN THE ARRAY ARE :"
110 DATA 10,27.7,13.2,40.1,-56.4,26.7,13.2,44.1,12.4,-26.7,100.6
120 LET C = 0
130 REM C EQUALS ZERO IF NO SWAPS ARE MADE
140 FOR I = 1 TO N-1
150 IF A(I)<=A(I+1) THEN 200
160 LET T = A(I)
170 LET A(I)=A(I+1)
180 LET A(I+1)=T
190 LET C = C+1
200 NEXT I
210 FOR K = 1 TO N:PRINT A(K);:NEXT K
220 PRINT
230 IF C>0 THEN 120
240 END
```

```
THE ORIGINAL ARRAY
 27.7  13.2  40.1 -56.4  26.7  13.2  44.1  12.4 -26.7  100.6
THE EXCHANGES IN THE ARRAY ARE :
 13.2  27.7 -56.4  26.7  13.2  40.1  12.4 -26.7  44.1  100.6
 13.2 -56.4  26.7  13.2  27.7  12.4 -26.7  40.1  44.1  100.6
-56.4  13.2  13.2  26.7  12.4 -26.7  27.7  40.1  44.1  100.6
-56.4  13.2  13.2  12.4 -26.7  26.7  27.7  40.1  44.1  100.6
-56.4  13.2  12.4 -26.7  13.2  26.7  27.7  40.1  44.1  100.6
-56.4  12.4 -26.7  13.2  13.2  26.7  27.7  40.1  44.1  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  40.1  44.1  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  40.1  44.1  100.6
```

The following program is a modified bubble sort. After an interchange, the checking procedure begins over at the first element of the array. It is unique in that a nested loop is not required, but also much slower.

```
10 REM A MODIFIED BUBBLE SORT
20 READ N
30 DIM A(100)
40 PRINT "THE ORIGINAL ARRAY"
50 FOR M = 1 TO N
60 READ A(M)
70 PRINT A(M);
80 NEXT M
90 PRINT
100 PRINT"THE EXCHANGES IN THE ARRAY ARE :"
110 DATA 10,27.7,13.2,40.1,-56.4,26.7,13.2,44.1,12.4,-26.7,100.6
120 FOR I = 1 TO N-1
130 IF A(I)<=A(I+1) THEN 180
140 LET T = A(I)
150 LET A(I)=A(I+1)
160 LET A(I+1)=T
165 FOR K = 1 TO N:PRINT A(K);:NEXT K:PRINT
170 GOTO 120
180 NEXT I
190 END
```

```
THE ORIGINAL ARRAY
 27.7  13.2  40.1 -56.4  26.7  13.2  44.1  12.4 -26.7  100.6
THE EXCHANGES IN THE ARRAY ARE :
 13.2  27.7  40.1 -56.4  26.7  13.2  44.1  12.4 -26.7  100.6
 13.2  27.7 -56.4  40.1  26.7  13.2  44.1  12.4 -26.7  100.6
 13.2 -56.4  27.7  40.1  26.7  13.2  44.1  12.4 -26.7  100.6
-56.4  13.2  27.7  40.1  26.7  13.2  44.1  12.4 -26.7  100.6
-56.4  13.2  27.7  26.7  40.1  13.2  44.1  12.4 -26.7  100.6
-56.4  13.2  26.7  27.7  40.1  13.2  44.1  12.4 -26.7  100.6
-56.4  13.2  26.7  27.7  13.2  40.1  44.1  12.4 -26.7  100.6
-56.4  13.2  26.7  13.2  27.7  40.1  44.1  12.4 -26.7  100.6
-56.4  13.2  13.2  26.7  27.7  40.1  44.1  12.4 -26.7  100.6
-56.4  13.2  13.2  26.7  27.7  40.1  12.4  44.1 -26.7  100.6
-56.4  13.2  13.2  26.7  27.7  12.4  40.1  44.1 -26.7  100.6
-56.4  13.2  13.2  26.7  12.4  27.7  40.1  44.1 -26.7  100.6
-56.4  13.2  13.2  12.4  26.7  27.7  40.1  44.1 -26.7  100.6
-56.4  13.2  12.4  13.2  26.7  27.7  40.1  44.1 -26.7  100.6
-56.4  12.4  13.2  13.2  26.7  27.7  40.1  44.1 -26.7  100.6
-56.4  12.4  13.2  13.2  26.7  27.7  40.1 -26.7  44.1  100.6
-56.4  12.4  13.2  13.2  26.7  27.7 -26.7  40.1  44.1  100.6
-56.4  12.4  13.2  13.2  26.7 -26.7  27.7  40.1  44.1  100.6
-56.4  12.4  13.2  13.2 -26.7  26.7  27.7  40.1  44.1  100.6
-56.4  12.4  13.2 -26.7  13.2  26.7  27.7  40.1  44.1  100.6
-56.4  12.4 -26.7  13.2  13.2  26.7  27.7  40.1  44.1  100.6
-56.4 -26.7  12.4  13.2  13.2  26.7  27.7  40.1  44.1  100.6
```

Computer Math

Name_____
Period_____

S A M P L E

Test I - Topic 9
Nested Loops and Sorting

1. What is the output of this program? 10 Pts.

```
10 PRINT "LENGTH", "WIDTH", "AREA"
20 FOR L = 1 TO 3
30 FOR W = 1 TO 3
40 PRINT L, W, L * W
50 NEXT W
60 NEXT L
70 END
```

2. Find the error in this program. 5 Pts.

```
10 FOR I = 1 TO 10
20 PRINT I
30 FOR J = 2 TO 20 STEP 2
40 LET K = I + J
50 PRINT K
60 NEXT I
70 NEXT J
80 END
```

3. The following program finds the largest number A(J) in a list of 10 numbers to be read in. 25 Pts.

_____ a. Line 10 blank should contain
A. A(J) B. A(I) C. J(10) D. A(K)

_____ b. Line 30 blank should contain
A. A(I) B. A(10) C. A(K) D. A(J)

_____ c. Line 60 blank should contain
A. 1,10 B. 2,10 C. J,10 D. 1,5-1

_____ d. Line 70 blank should contain
A. 100 B. 110 C. 90 D. 60

_____ e. Line 90 blank should contain
A. K B. I C. J+1 D. K-1

```
10 DIM _____
20 FOR I = 1 TO 10
30 DATA 3,5,8,0,-4,12,9,7,3,1
40 NEXT I
50 LET J = 10
60 FOR K = ____ TO ____
70 IF A(K) > A(J) THEN _____
80 GO TO 100
90 LET J = _____
100 NEXT K
110 PRINT A(J)
120 END
```

4. Arrange these #'s in descending order according to either sort program (sequential or bubble). Circle which method used. Copy the original array and rewrite it for each change. 15 Pts.

6 5 115 -7 11 -13

5. a. Which line # would be changed in #3 to find the smallest number in the list? _____ 10 Pts.

   b. Rewrite the line.

   _____

6. Write a program to print the integers 2, 4, 6, 8, and 10 exactly 3 times. 20 Pts.

7. Write the 3 program steps necessary to switch the values of this pair of variables: M(1) and M(10) 15 Pts.

Computer Math

ANSWER KEY

Name_____
Period_____

SAMPLE

Test I - Topic 9
Nested Loops and Sorting

1. What is the output of this program? 10 Pts.

```
10 PRINT "LENGTH", "WIDTH", "AREA"
20 FOR L = 1 TO 3
30 FOR W = 1 TO 3
40 PRINT L, W, L * W
50 NEXT W
60 NEXT L
70 END
```

LENGTH	WIDTH	AREA
1	1	1
1	2	2
1	3	3
2	1	2
2	2	4
2	3	6
3	1	3
3	2	6
3	3	9

2. Find the error in this program. 5 Pts.

```
10 FOR I = 1 TO 10
20 PRINT I
30 FOR J = 2 TO 20 STEP 2
40 LET K = I + J
50 PRINT K
60 NEXT I
70 NEXT J
80 END
```

SWITCH

3. The following program finds the largest number A(J) in a list of 10 numbers to be read in. 25 Pts.

C a. Line 10 blank should contain
A. A(J) B. A(I) C. J(10) D. A(K)

A b. Line 30 blank should contain
A. A(I) B. A(10) C. A(K) D. A(J)

D c. Line 60 blank should contain
A. 1,10 B. 2,10 C. J,10 D. 1,5-1

C d. Line 70 blank should contain
A. 100 B. 110 C. 90 D. 60

A e. Line 90 blank should contain
A. K B. I C. J+1 D. K-1

```
10 DIM _____
20 FOR I = 1 TO 10
30 DATA 3,5,8,0,-4,12,9,7,3,1
40 NEXT I
50 LET J = 10
60 FOR K = ____ TO ____
70 IF A(K) > A(J) THEN _____
80 GO TO 100
90 LET J = _____
100 NEXT K
110 PRINT A(J)
120 END
```

4. Arrange these #'s in descending order according to either sort program (sequential or bubble). Circle which method used. Copy the original array and rewrite it for each change. 15 Pts.

6 5 115 -7 11 -13

5. a. Which line # would be changed in #3 to find the smallest number in the list? 70 10 Pts.

   b. Rewrite the line.

IF A(K) < A(J) THEN 90

6. Write a program to print the integers 2, 4, 6, 8, and 10 exactly 3 times. 20 Pts.

```
10 FOR J = 1 TO 3
20 FOR I = 2 TO 10 STEP 2
30 PRINT I
40 NEXT I
50 NEXT J
```

7. Write the 3 program steps necessary to switch the values of this pair of variables: M(1) and M(10) 15 Pts.

```
LET T=M(1)                  LET T=M(10)
LET M(1)=M(10)     or       LET M(10)=M(1)
LET M(10)=T                 LET M(1)=T
```

SEQUENTIAL

6	5	115	-7	11	-13
115	5	6	-7	11	-13
115	6	5	-7	11	-13
115	11	5	-7	6	-13
115	11	6	-7	5	-13
115	11	6	5	-7	-13

BUBBLE

6	5	115	-7	11	-13
6	115	5	-7	11	-13
115	6	5	-7	11	-13
115	6	5	11	-7	-13
115	6	11	5	-7	-13
115	11	6	5	-7	-13

TOPIC TEN: ALPHAMERIC OUTPUT AND INPUT

Performance Objectives

The student will -

A14 - Write BASIC programs which utilize string variables.

A14a - Write BASIC programs which utilize the command CLEAR.

A17(n) - Alphabetize a list of words.

References

Textbook: Pages 295-317

Appendix B, pages 453-454

Objective A14: The student will write BASIC programs which utilize string variables.

Textbook References: Pages 295-298
Pages 300-304

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
NOTE: Check system used to determine whether quotation marks are necessary or useful in data for string variables. In most systems, strings of one word can omit quotation marks in the data statement. For strings containing commas and more than one word, however, quotation marks are required. String variables are important in programming especially in making computers useful for other than straight mathematical purposes. A string is alphanumeric or numeric information that may be any combination of characters. A string variable is a variable used to represent a string. String can be used in INPUT, PRINT, and READ-DATA statements.	Textbook Oral Exercise One, p. 298, #1-4
This develops use of string variables in IF-THEN statements. Be sure students understand use of " = " with strings.	Textbook Oral Exercise Two, pp. 303-304 #1-4 Written PROBLEM SET TEN pp. 316-317 #1-6

Objective A14: String variables (continued)

A17(n): The student will alphabetize a list of words.

Textbook References: Pages 306-309, 454-454

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
This section combines string variables and subscripts. Check system used to determine exact limitations on types of string variables accepted. With this material traces again will be very helpful. Objective A17(n): Upon completing this chapter, students should be ready to do alphabetic sorts. (Pages 453-454 of textbook) Go over this carefully through use of trace.	Textbook Oral Exercise Three, pp. 308-309 #1-3 Objective Activity, pp. 311-313 #1-7 Written PROBLEM SET TEN pp. 317-318 #7-8 Programs PROBLEM SET TEN p. 818 #9-12 Extra Programs, including an alphabetic sort

Objective A14a: The student will write BASIC programs which utilize the command CLEAR.

Textbook Reference: None

TEACHING SUGGESTIONS

The command "CLEAR" sets the values of all variables to zero, including the index of a FOR...NEXT loop.

```
 70 A=3:B(1,2)=4
 80 PRINT A;B(1,2);
 90 CLEAR
100 PRINT A;B(1,2)
```

HAS OUTPUT: 3 4 0 0

Clear N, where N is a positive integer, does two things. First, it acts like the CLEAR command. Second, it changes the size of 'string space' to N characters. When a program manipulates long strings it may be necessary to increase the size of string space in order to avoid an 'out of string space' error.

STUDENT ACTIVITIES IN THIS GUIDE

Have student try the following programs:

1.
```
10 A$ = "A":B$ = ""
20 FOR I = 1 TO N
30 B$ = B$ + A$
40 PRINT B$
50 NEXT I
   for N = 10, 50, 500
```

Repeat #1 with statement
```
5 CLEAR 100
```

2.
```
10 A=7:A$ = "APPLE":A(Ø) = 1
20 CLEAR
30 B = 5
40 PRINT A, A$, A(Ø), B
```

Computer Math

Name_____
Period_____

SAMPLE TEST

Test on Topic 10

Tell which of the following are valid string variables. (Yes or No) 30 Pts.

_____ 1. M2$

_____ 2. J$(3)

_____ 3. V$3

_____ 4. 6$Q

_____ 5. W$

_____ 6. ST

Construct read statements to match the following DATA statements. Choose your own variable names. 10 Pts.

7. DATA 4, "JOHN DOE", 42, 750

_____

8. DATA SMITH, 1234 MAIN, HOUSTON, TEXAS, 77074

_____

Find and correct all errors. 25 Pts.

9. 10 LET N$ = NO

10. 40 IF A$ = YES THEN 60

11. 50 INPUT "YES"

12. 20 READ A,B, "JONES"

13. 10 DIM A$ 10

14. Determine the entire printout of the following program. 15 Pts

```
 5 DATA 45, SMITH, 25, REED, 40, JONES, 41, DALY, 35, FORD, 50, JOHNSON
10 FOR I = 1 TO 6
20 READ X(I), A$(I)
30 NEXT I
40 FOR I = 1 TO 6
50 IF X(I) < = 40 THEN 70
60 PRINT A$(I), X(I)
70 NEXT I
80 END
```

15. Write a statement to randomly generate one of the 26 letters of the alphabet, stored in an array A$(I). 5 Pts.

_____

16. Fill in the missing line. This program will print "YES", "NO", or "MAYBE" - randomly generated. It is in a continuous loop.

```
 5 FOR I = 1 TO 3
10 READ A$(I)
20 NEXT I
25 DATA YES, NO, MAYBE
30
40 PRINT A$(I)
50 GO TO 30
```

ANSWER KEY

Computer Math

Name_____
Period_____

SAMPLE TEST

Test on Topic 10

Tell which of the following are valid string variables. (Yes or No) 30 Pts.

Yes 1. M2$

Yes 2. J$(3)

No 3. V$3

No 4. 6$Q

Yes 5. W$

No 6. ST

Construct read statements to match the following DATA statements. Choose your own variable names. 10 Pts.

7. DATA 4, "JOHN DOE", 42, 750

READ C, N$, H, R

8. DATA SMITH, 1234 MAIN, HOUSTON, TEXAS, 77074

READ N$, A$, C$, S$, Z

Find and correct all errors. 25 Pts.

9. 10 LET N$ = "NO"

10. 40 IF A$ = "YES" THEN 60

11. 50 INPUT "YES" 50 INPUT; A$

12. 20 READ A,B, "JONES" N$

13. 10 DIM A$ (10)

14. Determine the entire printout of the following program. 15 Pts

```
 5 DATA 45, SMITH, 25, REED, 40, JONES, 41, DALY, 35, FORD, 50, JOHNSON
10 FOR I = 1 TO 6
20 READ X(I), A$(I)
30 NEXT I
40 FOR I = 1 TO 6
50 IF X(I) < = 40 THEN 70
60 PRINT A$(I), X(I)
70 NEXT I
80 END
```

```
SMITH   45
DALY    41
JOHNSON 50
```

15. Write a statement to randomly generate one of the 26 letters of the alphabet, stored in an array A$(I). 5 Pts.

```
LET I = INT(RND(Ø) * 26 + 1)
```

16. Fill in the missing line. This program will print "YES", "NO", or "MAYBE" - randomly generated. It is in a continuous loop.

```
 5 FOR I = 1 TO 3
10 READ A$(I)
20 NEXT I
25 DATA YES, NO, MAYBE
30 LET I = INT(RND(Ø)*3+1)
40 PRINT A$(I)
50 GO TO 30
```

Indicate which operation is to be performed first:

_____ 1. 10 LET R = A * (B + C)
_____ 2. 10 LET Q = A * B↑2
_____ 3. 10 LET S1 = B - V * D
_____ 4. 10 LET D = A - X + Z
_____ 5. 10 LET C = X + Y - Z

Correct any errors in the following BASIC statements:

1. 10 LET X + Y = Q
2. END
3. 10 READ C;D
4. 45 GO TO TEN
5. 100 DATA 6,8: 10,12
6. 50 PRINT X " = X"
7. 20 LET Z↑2 = X↑2 - Y↑2

Which of the following are valid IF ... THEN statements?

_____ 1. 30 IF T / D = R GO TO 50
_____ 2. 30 IF X = 3, THEN 12
_____ 3. 30 IF R + 5 = Q = S THEN 80
_____ 4. IF A + B < C - D THEN 60

Which of the following are valid PRINT statements?

_____ 1. 80 PRINT "AREA" = A
_____ 2. 100 PRINT X, Y; Z
_____ 3. 15 PRINT
_____ 4. 20 PRINT "(" X ",Y")"
_____ 5. 100 PRINT "SIDE = " S " PERIM = " 4 * S
_____ 6. 20 PRINT S "= SUM"

The following program test whether a number is outside the range $2 \leq A \leq 10$.
Circle the correct answer.

```
 10 LET S = 1
 20 READ A
 30 IF A > 12 THEN _____
 40 IF A < THEN _____
 50 IF S > = 20 THEN _____
 60 LET S = _____
 70 GO TO _____
 80 PRINT A, "OUTSIDE RANGE"
 90 _____
100 DATA ---(20 numbers)
110 END
```

1. Line 30 blank should contain
   A. 110 B. 80 C. 70 D. 60
2. Line 40 blank should contain
   A. 60 B. 70 C. 110 D. 80
3. Line 50 blank should contain
   A. 60 B. 70 C. 110 D. 80
4. Line 60 blank should contain
   A. S + 1 B. S + 2 C. S + A D. A
5. Line 70 blank should contain
   A. 100 B. 60 C. 20 D. 10
6. Line 90 should contain
   A. GO TO 10 B. GO TO 20
   C. GO TO 50 D. GO TO 60

True-False

_____ 1. DIM stands for dimension.
_____ 2. DIM A(50) means that A(I) cannot store values that are larger than 50. (That is, A(3) could not equal 58.)
_____ 3. If B = (38 24 97 88 102) then the DIM statement needed for the array is DIM B (102).
_____ 4. DIM X(20) means that values can be stored in X(1) → X(20).
_____ 5. The number of values in a list may be less than the number shown in the DIM statement but never more.

Indicate which operation is to be performed first:

+	1.	10 LET R = A * (B + C)
↑	2.	10 LET Q = A * B↑2
*	3.	10 LET S1 = B - V * D
-	4.	10 LET D = A - X + Z
+	5.	10 LET C = X + Y - Z

Correct any errors in the following BASIC statements:

1.	10 LET X + Y = Q	10 LET Q = X + Y
2.	END	20 END
3.	10 READ C;D	10 READ C, D
4.	45 GO TO TEN	45 GO TO 10
5.	100 DATA 6,8: 10,12	100 DATA 6, 8, 10, 12
6.	50 PRINT X " = X"	OK
7.	20 LET Z↑2 = X↑2 - Y↑2	20 LET Z = SQR (X↑2 - Y↑2)

Which of the following are valid IF ... THEN statements?

NO	1.	30 IF T / D = R GO TO 50
NO	2.	30 IF X = 3 THEN 12
NO	3.	30 IF R + 5 = Q = S THEN 80
YES	4.	IF A + B < C - D THEN 60

Which of the following are valid PRINT statements?

NO	1.	80 PRINT "AREA" = A
YES	2.	100 PRINT X, Y; Z
YES	3.	15 PRINT
NO	4.	20 PRINT "(" X ",Y")"
YES	5.	100 PRINT "SIDE = " S " PERIM = " 4 * S
YES	6.	20 PRINT S "= SUM"

The following program test whether a number is outside the range $2 \leq A \leq 10$.
Circle the correct answer.

```
 10 LET S = 1
 20 READ A
 30 IF A > 12 THEN _____
 40 IF A < THEN _____
 50 IF S > = 20 THEN _____
 60 LET S = _____
 70 GO TO _____
 80 PRINT A, "OUTSIDE RANGE"
 90 _____
100 DATA ---(20 numbers)
110 END
```

1. Line 30 blank should contain
   A. 110 B. 80 C. 70 D. 60
2. Line 40 blank should contain
   A. 60 B. 70 C. 110 D. 80
3. Line 50 blank should contain
   A. 60 B. 70 C. 110 D. 80
4. Line 60 blank should contain
   A. S + 1 B. S + 2 C. S + A D. A
5. Line 70 blank should contain
   A. 100 B. 60 C. 20 D. 10
6. Line 90 should contain
   A. GO TO 10 B. GO TO 20
   C. GO TO 50 D. GO TO 60

True-False

T	1.	DIM stands for dimension.
F	2.	DIM A(50) means that A(I) cannot store values that are larger than 50. (That is, A(3) could not equal 58.)
F	3.	If B = (38 24 97 88 102) then the DIM statement needed for the array is DIM B (102).
T	4.	DIM X(20) means that values can be stored in X(1) → X(20).
T	5.	The number of values in a list may be less than the number shown in the DIM statement but never more.

TOPIC ELEVEN: TWO-DIMENSIONAL ARRAYS

Performance Objectives:

The student will -

A14a - Write BASIC programs which utilize double subscripts.

A17a(e) - Be able to write a program that will print an N x N array of numbers and check whether it is a magic square.

B6 - Write BASIC programs which utilize MAT commands.

Reference:

Textbook, pp. 319 - 354

Objective A14a: The student will write BASIC programs which utilize double subscripts.

B6: The student will write BASIC programs which utilize MAT commands.

Textbook Reference: Pages 319-331

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
This first section is primarily introduction to numerical arrays and their designations. The second section introduces DIM, MAT READ, and MAT PRINT. Carefully clarify difference between use of READ within nested FOR-NEXT loops and use of MAT READ to input data into two-dimensional array and between use of PRINT within nested loops and use of MAT PRINT to output data.	Textbook Oral Exercise One, pp. 324-325 #1-3 Oral Exercise Two, pp. 331 #1-5 Objective Activity, pp. 343-344 #1-7 Written PROBLEM SET ELEVEN pp. 350-352 #1-10 Program PROBLEM SET ELEVEN pp. 353 #11

Objective A14a: continued

A17a(e): The student will print an NXN array and check whether it is a magic square.

Textbook Reference: Pages 333-354

TEACHING SUGGESTIONS

The remainder of Topic Eleven is made up of examples of uses of two-dimensional arrays.

Once again the use of traces is a real aid to understanding these programs.

Objective A17a(e):

The program on p. 336 of the textbook prints and checks whether an array is a magic square. Program A17a(e) of the Program Appendix takes a similar program and extends it to check for Latin squares as well.

The flowchart (Fig. 11.14, p. 335) and program (Fig. 11.15, p. 336) to find the 2nd diagonal of a magic square could be altered in one of these ways:

```
240 LET N=4 : LET S = 0
250 FOR I = 1 TO 4
260 LET S = S + L (I,N)
270 LET N = N - 1
300 NEXT I
310 PRINT "SECOND DIAGONAL SUM IS"; S
```

```
240 LET N = 4: LET S = 0
250 FOR I = 1 TO 4
260 LET S = S + L(I,N+1-I)
300 NEXT I
310 PRINT "SECOND DIAGONAL SUM IS"; S
```

```
240 LET N = 1: LET S = 0
250 FOR I = 4 TO 1 STEP-1
260 LET S = S + L (I,K)
270 K = K + 1
300 NEXT I
310 PRINT "SECOND DIAGONAL SUM IS"; S
```


STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral
Exercise Three, pp. 338-340, #1-5

Objective Activity, pp. 344-345, #8-11

Programs
PROBLEM SET ELEVEN
pp. 353-354, #12-16

SAMPLE TEST

Computer Math

Test on Topic 11

Name_____
Period_____

1. Write dimension statements for the following arrays: (10 Pts.)

   a. A array _____ b. B array _____

6	8	-2	12
0	4	3	-8
16	12	15	9

34	28	15	20
18	7	9	37

2. In the A array, determine the value of the following. (15 Pts.)

   a. A(3,2) _____
   b. A(2,3) _____
   c. A(1,2) _____
   d. A(2,1) + A(3,3) = _____
   e. A(3,2) - A(2,4) = _____

3. What is the array name for each of the following numbers in the B array? (15 Pts.)

   a. 7 _____
   b. 37 _____
   c. 20 _____
   d. 18 _____
   e. 9 _____

4. Print the results of this program: (15 Pts.)

```
10 DIM A(3,3)
20 FOR I = 1 TO 3
30 FOR J = 1 TO 3
40 LET A (I,J) = I+J
50 PRINT A (I,J)
60 NEXT J
70 PRINT
80 NEXT I
90 END
```

5. Print the results of this partial program if the array is as follows: (5 Pts.)

```
   8  1  6
   3  5  4
   2  9  7

10 LET L = D(1,1)
20 FOR I = 1 TO 3
30 FOR J = 1 TO 3
40 IF L < D(I,J) THEN 60
50 LET L = D(I,J)
60 NEXT J
70 NEXT I
80 PRINT L
```

6. Write the formula for summing the rows in A(a 4x4 matrix). (5 Pts.)

```
10 FOR I = 1 TO 4
20 N = 0
30 FOR J = 1 TO 4
40 LET N = _____________
50 NEXT J
60 PRINT "ROW"; I; "SUM IS"; N
70 NEXT I
80 END
```

7. Print the results of this partial program. Assume that A(4,4) has been properly read and stored in the computer. (5 Pts.)

```
A =  16    2    5   13
      8   21   10    8
      9    7    6   13
      4   14   15    4
```

```
 70 K = 4
 75 S = 0
 80 FOR I = 1 TO 4
 90 S = S + A(K,I)
100 K = K - 1
110 NEXT I
120 PRINT S
```

8. Write a program to read this matrix, print it, and find its trace (the sum of the main diagonal). (30 Pts.)

```
L =  6   5   3
     2   1   8
     7   9   0
```

ANSWER KEY

SAMPLE TEST

Computer Math

Test on Topic 11

Name _____
Period _____

1. Write dimension statements for the following arrays: (10 Pts.)

   a. A array DIM A (3,4)

   b. B array DIM B (2,4)

6	8	-2	12
0	4	3	-8
16	12	15	9

34	28	15	20
18	7	9	37

2. In the A array, determine the value of the following. (15 Pts.)

   a. A(3,2) 12
   b. A(2,3) 3
   c. A(1,2) 8
   d. A(2,1) + A(3,3) = 15
   e. A(3,2) - A(2,4) = 20

3. What is the array name for each of the following numbers in the B array? (15 Pts.)

   a. 7 B(2,2)
   b. 37 B(2,4)
   c. 20 B(1,4)
   d. 18 B(2,1)
   e. 9 B(2,3)

4. Print the results of this program: (15 Pts.)

```
10 DIM A(3,3)
20 FOR I = 1 TO 3
30 FOR J = 1 TO 3
40 LET A (I,J) = I+J
50 PRINT A (I,J)
60 NEXT J
70 PRINT
80 NEXT I
90 END
```

Array A

2	3	4
3	4	5
4	5	6

5. Print the results of this partial program if the array is as follows: (5 Pts.)

8	1	6
3	5	4
2	9	7

```
10 LET L = D(1,1)
20 FOR I = 1 TO 3
30 FOR J = 1 TO 3
40 IF L < D(I,J) THEN 60
50 LET L = D(I,J)
60 NEXT J
70 NEXT I
80 PRINT L
```

Output 1

6. Write the formula for summing the rows in A(a 4x4 matrix). (5 Pts.)

```
10 FOR I = 1 TO 4
20 N = 0
30 FOR J = 1 TO 4
40 LET N =    N + A(I,J)
50 NEXT J
60 PRINT "ROW"; I; "SUM IS"; N
70 NEXT I
80 END
```

7. Print the results of this partial program. Assume that A(4,4) has been properly read and stored in the computer. (5 Pts.)

```
A =  16    2    5   13
      8   21   10    8
      9    7    6   13
      4   14   15    4
```

```
 70 K = 4
 75 S = 0
 80 FOR I = 1 TO 4
 90 S = S + A(K,I)      Output    34
100 K = K - 1
110 NEXT I
120 PRINT S
```

8. Write a program to read this matrix, print it, and find its trace (the sum of the main diagonal). (30 Pts.)

```
L =  6    5    3
     2    1    8
     7    9    0
```

```
 10 DIM  L(3,3)
 20 FOR I = 1 TO 3
 30 FOR J = 1 TO 3
 40 READ L(I,J)
 50 PRINT L(I,J);
 60 NEXT J
 70 PRINT
 80 NEXT I
 90 DATA 6,5,3,2,1,8,7,9,0
100 FOR I = 1 TO 3
110 S = S + A(I,I)
120 NEXT I
130 PRINT S
140 END
```

TOPIC TWELVE: FUNCTIONS AND SUBROUTINES

Performance Objectives:

The student will -

A17(c) - Generate prime numbers less than 300.

A17(d) - List the integral divisors of any number less than 300.

A17(h) - Print all the prime factors of a given integer less than 500.

A17(o) - Write BASIC programs which will plot the curve of simple functions.

B5 - Write BASIC programs which utilize user-defined functions and and subroutines.

Reference:

Textbook, pp. 355-387

Objective B5: The student will write BASIC programs which utilize user-defined functions and subroutines.

A17(d): The student will write BASIC programs which will plot the curves of simple functions.

Textbook References: PP. 355-360, 454-457

TEACHING SUGGESTIONS

Be sure students understand method of defining functions and do not confuse arguments and subscripts.

Once defined, functions can be used to assign values to variables.

They can be used in decision statements.

Pages 454-457 show a use of defined functions in plotting curves. The use of defined function makes the general program more flexible.

The sample programs for using subroutines, Fig. 12.16 and Fig. 12.18, are not good examples of subroutines.* An alternative might be the following--from the flowchart, Fig. 12.7--notice also that the answers for #2, Ex. Two on p. 369 are incorrect. #2 should read:

2. The values would be
   A(1) = 15 A(4) = 12 A(9) = 7
   A(12) = 4 A(7) = 9 A(15) = 1

*These programs can be written efficiently without subroutines. Subroutines should not be used just for the sake of using subroutines.

Leroy Finkel and Jerald R. Brown in DATA FILE Programming in BASIC (New York: John Wiley & Sons, Inc., 1981) state (p. 9):

"Technically, you need use a subroutine only to avoid duplicating the same program statements in two or more places in your program. A subroutine should be called from MORE than one place in your program."

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral
Exercise One, pp. 359-360 #1-4

Objective Activity, pp. 378-379 #1-5

Written
PROBLEM SET TWELVE
pp. 385 #1-6

Program
Plot the graph of a function similar to the example on page 457.

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
...and (p. 14): to save space... "Use GOTO, not GOSUB, for a routine accessed from only one place in a program."	

185

```
10 DIM A(15)
20 N = 15
30 FOR I = 1 TO N
40 LET A(I) = I
50 NEXT I
60 GOSUB 160
70 LET K = INT(N/2)
80 FOR I = 1 TO K STEP 3
90 LET J = N + 1 -I
100 LET T = A(I)
110 LET A(I) = A(J)
120 LET A(J) = T
130 NEXT I
140 GOSUB 160
150 GOTO 210
160 FOR I =1 TO N
170 LPRINT "A("I") =" A(I)
180 NEXT I
190 LPRINT:LPRINT
200 RETURN
210 END
```

```
A( 1 ) = 1
A( 2 ) = 2
A( 3 ) = 3
A( 4 ) = 4
A( 5 ) = 5
A( 6 ) = 6
A( 7 ) = 7
A( 8 ) = 8
A( 9 ) = 9
A( 10 ) = 10
A( 11 ) = 11
A( 12 ) = 12
A( 13 ) = 13
A( 14 ) = 14
A( 15 ) = 15
```

```
A( 1 ) = 15
A( 2 ) = 2
A( 3 ) = 3
A( 4 ) = 12
A( 5 ) = 5
A( 6 ) = 6
A( 7 ) = 9
A( 8 ) = 8
A( 9 ) = 7
A( 10 ) = 10
A( 11 ) = 11
A( 12 ) = 4
A( 13 ) = 13
A( 14 ) = 14
A( 15 ) = 1
```

Objective B5: continued

A17(o): continued

Textbook Reference: Pp. 362-386

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
One important purpose of subroutines is convenience. In real programs subroutines are used when the same process must be repeated at different places within the program. Subroutines eliminate the necessity of continually rewriting the same statements. Format is crucial. Carefully go over an individual subroutine. Then follow through an entire program containing subroutine(s). Use a trace. Subroutines are entered only from GOSUB--and left only from RETURN.	Textbook Oral Exercise Two, pp. 366-368 #1-4 Objective Activity, p. 380 #6-7 Written PROBLEM SET TWELVE pp. 385-386 #7-10

187

Objective B5: continued

A17(c): The student will generate prime numbers less than 300.

A17(d): The student will list the integral divisors of any integer less than 100.

A17(h): The student will print all the prime factors of a given integer less than 500.

Textbook Reference: Pp. 370-387

TEACHING SUGGESTIONS

The last section of this topic is composed of many examples of programs using functions and subroutines.

Go over enough of these to be certain students have an understanding. Do traces.

On pp. 374 and 375 are listed programs which illustrate objectives A17(c) (prime numbers) and A17(d) (integral divisors). Simple programs can also be written without the use of subroutines. (See Program Appendix.) It is suggested that the latter approach be taken during first semester.

Objective A17(h) provides a good application of using prime numbers. See Program A17(h) in the Program Appendix.

Supplement 5.1 (Graphs of Three-Dimensional Functions) is an example of the combined use of PRINT USING, defined functions, and subroutines. Some students may find these interesting and useful.

On page 363, the idea of multiple returns for one subroutine is introduced. This use of return is not universally accepted in programming.

STUDENT ACTIVITIES IN THIS GUIDE

Textbook

Oral

Exercise Three,
p. 376 #1-4

Objective Activity,
p. 381 #8-12

Written

PROBLEM SET TWELVE
p. 386 #11-14

Programs

PROBLEM SET TWELVE
p. 387 #15-20

```
 10 DEF FNZ(X,Y)=X^2-Y^2
100 PRINT "REPRESENTATIONS OF THREE-DIMENSIONALFUNCTIONS"
110 PRINT
120 PRINT
130 PRINT "REPRESENTATIONS OF F(X,Y) "
140 PRINT "FOR X FROM ? TO ?"
150 INPUT A,B
160 PRINT "AND FOR Y FROM ? TO ?"
170 INPUT C,D
180 GOSUB 5000
190 FOR J=0 TO 36
200 FOR I=0 TO 59
210 GOSUB 2000
215 PRINT Q$(K);
330 NEXT I
335 I=60
336 PRINT Q$(K)
340 Y=Y-(D-C)/36
350 NEXT J
360 X$=" X-RANGE IS ####.## TO ####.##"
370 Y$="Y-RANGE IS ####.## TO ####.##"
380 R$=" # REPRESENTS A RANGE OF ####.## TO ####.##"
390 PRINT
400 PRINT USING X$,A,B
410 PRINT USING Y$,C,D
420 FOR K=0 TO 9
430 U=K*(T-S)/10
440 V=(K+1)*(T-S)/10
450 PRINT USING R$,K,U,V
460 NEXT K
470 GOTO 110
480 END
2000 X=A+I*(B-A)/60
2010 Y=D-J*(D-C)/36
2020 Z=FNZ(X,Y)
2030 FOR L=0 TO 8 STEP 2
2040 Q$(L)=" "
2050 NEXT L
2060 Q$(1)="1" : Q$(3)="3": Q$(5)="5": Q$(7)="7": Q$(9)="9"
2110 K=INT(10*FNZ(X,Y)/(T-S))
2120 IF K<10 THEN 2150
2130 K=9
2140 GOTO 2170
2150 IF K>=0 THEN 2170
2160 K=0
2170 RETURN
5000 T=-1E10
5010 S=1E10
5020 FOR Y=C TO D STEP (D-C)/36
5030 FOR X=A TO B STEP (B-A)/60
5040 Z=FNZ(X,Y)
5050 IF S<Z THEN 5090
5060 S=Z
5070 G=X
5080 E=Y
5090 IF T>Z THEN 5130
5100 T=Z
5110 H=X
5120 F=Y
5130 NEXT X
5140 NEXT Y
5150 RETURN
```

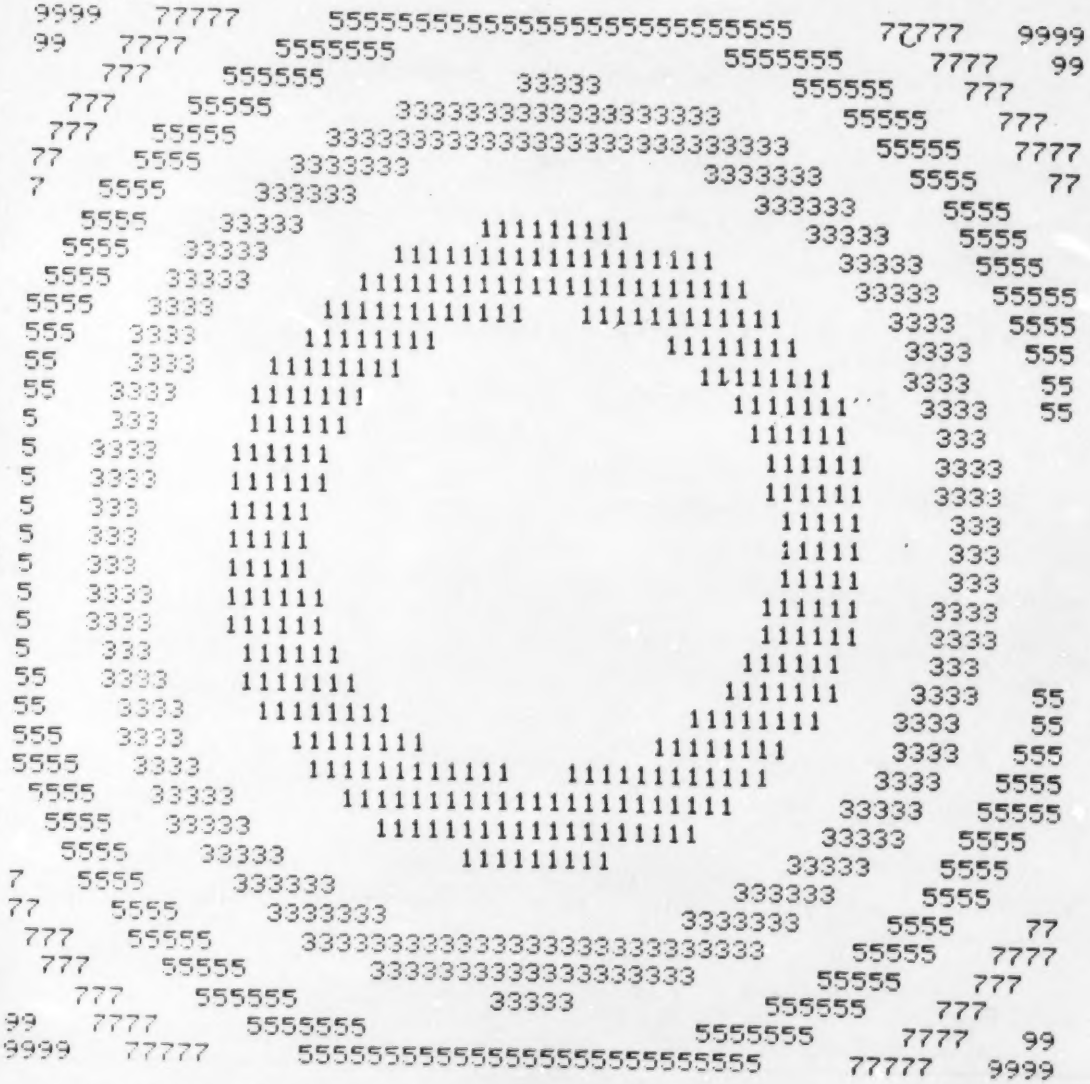
```
 10 DEF FNZ(X,Y) = X*Y*E^X/(1+Y^2)
 *
 RUN
REPRESENTATIONS OF THREE-DIMENSIONALFUNCTIONS


REPRESENTATIONS OF F(X,Y)
FOR X FROM ? TO ?
?1,5
AND FOR Y FROM ? TO ?
?1,5
     1111111111111111111                    33333333333333333
     1111111111111111111                   333333333333333333
    11111111111111111111                  3333333333333333333
    11111111111111111111                 33333333333333333333
   111111111111111111111                33333333333333333333
   111111111111111111111               3333333333333333333
   111111111111111111111              333333333333333333
  1111111111111111111111             33333333333333333
  1111111111111111111111            3333333333333333
 11111111111111111111111           333333333333333
 11111111111111111111111          33333333333333
 11111111111111111111111         3333333333333
11111111111111111111111         333333333333333                      55
11111111111111111111111        33333333333333                      5555
11111111111111111111111       333333333333333                     55555
11111111111111111111111      3333333333333333                    555555
 1111111111111111111111     33333333333333333                   5555555
 1111111111111111111111    333333333333333333                  55555555
 1111111111111111111111   3333333333333333333                 555555555
 1111111111111111111111  33333333333333333333               5555555555
 1111111111111111111111 333333333333333333333             55555555555
 1111111111111111111111 333333333333333333333            555555555555
 1111111111111111111111 333333333333333333333           5555555555555
 1111111111111111111111 333333333333333333333          55555555555555
 1111111111111111111111 333333333333333333333         555555555555555
 1111111111111111111111 333333333333333333333        5555555555555555
 1111111111111111111111 333333333333333333333       55555555555555555
 1111111111111111111111 333333333333333333333      555555555555555555
 1111111111111111111111 333333333333333333333     5555555555555555555
 1111111111111111111111 333333333333333333333    55555555555555555555
 1111111111111111111111 333333333333333333333   555555555555555555555
 1111111111111111111111 333333333333333333333  5555555555555555555555
 1111111111111111111111 333333333333333333333 55555555555555555555555
 1111111111111111111111 333333333333333333333 55555555555555555555555
 1111111111111111111111 333333333333333333333 55
```

```
10 DEF FNZ(X,Y) = COS(X)/(1+Y↑2)
 *RUN
REPRESENTATIONS OF THREE-DIMENSIONALFUNCTIONS


REPRESENTATIONS OF F(X,Y)
FOR X FROM ? TO ?
?0,1
AND FOR Y FROM ? TO ?
?0,1
                                  555555555555555             3333
7777777                             5555555555555              33
777777777777777                       55555555555
77777777777777777777                    555555555555
7777777777777777777777777                 55555555555
777777777777777777777777777777              5555555555
          77777777777777777777                5555555555
                  777777777777777               5555555555
                      77777777777777              555555555
                         7777777777777             55555555
99                          777777777777            55555555
99999999999999                 77777777777          555555555
9999999999999999999               7777777777         5555555
999999999999999999999999            7777777777       555555
99999999999999999999999999999         777777777       55555
9999999999999999999999999999999999      77777777       5555
999999999999999999999999999999999999999   77777777      555
99999999999999999999999999999999999999999  77777777      55
999999999999999999999999999999999999999999   7777777
9999999999999999999999999999999999999999999999  7777777
99999999999999999999999999999999999999999999999  7777777
999999999999999999999999999999999999999999999999  7777777
9999999999999999999999999999999999999999999999999  777777
99999999999999999999999999999999999999999999999999  7777777
999999999999999999999999999999999999999999999999999  777777
9999999999999999999999999999999999999999999999999999  777777
99999999999999999999999999999999999999999999999999999  77777
999999999999999999999999999999999999999999999999999999  77777
9999999999999999999999999999999999999999999999999999999  7777
99999999999999999999999999999999999999999999999999999999  7777
999999999999999999999999999999999999999999999999999999999  7777
9999999999999999999999999999999999999999999999999999999999  7777
99999999999999999999999999999999999999999999999999999999999  7777

 X-RANGE IS     0.00 TO     1.00
Y-RANGE IS     0.00 TO     1.00
 0 REPRESENTS A RANGE OF     0.00 TO     0.07
 1 REPRESENTS A RANGE OF     0.07 TO     0.15
 2 REPRESENTS A RANGE OF     0.15 TO     0.22
 3 REPRESENTS A RANGE OF     0.22 TO     0.29
 4 REPRESENTS A RANGE OF     0.29 TO     0.36
 5 REPRESENTS A RANGE OF     0.36 TO     0.44
 6 REPRESENTS A RANGE OF     0.44 TO     0.51
 7 REPRESENTS A RANGE OF     0.51 TO     0.58
 8 REPRESENTS A RANGE OF     0.58 TO     0.66
 9 REPRESENTS A RANGE OF     0.66 TO     0.73
```

```
10 DEF FNZ(X,Y) = COS(X*Y)
 *

  RUN
REPRESENTATIONS OF THREE-DIMENSIONAL FUNCTIONS


REPRESENTATIONS OF F(X,Y)
FOR X FROM ? TO ?
?-2,2
AND FOR Y FROM ? TO ?
?-2,2
                      11  33     5     33  11
                      1  33      5      33  1
                     11  33      5      33  11
                    11  333      5      333  11
                   11   33       5       33   11
                  11   333       5       333   11
                 11   333        5        333   11
                11   3333        5        3333   11
              111   3333         5         3333   111
            111    3333          5          3333    111
          1111    33333           5           33333    1111
      1111      33333              5              33333      1111
1111          333333                5                 333333          1111
     33333333                       5                    33333333
333333333                           5                        333333333
33                                  5                                33
                                    5
                                    5
5555555555555555555555555555555555555555555555555555555555555555555
                                    5
                                    5
33                                  5                                33
333333333                           5                        333333333
     33333333                       5                    33333333
1111          333333                5                 333333          1111
      1111      33333              5              33333      1111
          1111    33333           5           33333    1111
            111    3333          5          3333    111
              111   3333         5         3333   111
                11   3333        5        3333   11
                 11   333        5       333   11
                  11   333       5      333   11
                   11   33       5       33   11
                    11  333      5      333  11
                     11  33      5      33  11
                      1  33      5      33  1
                      11  33     5     33  11

 X-RANGE IS   -2.00 TO    2.00
 Y-RANGE IS   -2.00 TO    2.00
 0 REPRESENTS A RANGE OF     0.00 TO     0.20
 1 REPRESENTS A RANGE OF     0.20 TO     0.40
 2 REPRESENTS A RANGE OF     0.40 TO     0.60
 3 REPRESENTS A RANGE OF     0.60 TO     0.80
 4 REPRESENTS A RANGE OF     0.80 TO     1.00
 5 REPRESENTS A RANGE OF     1.00 TO     1.20
 6 REPRESENTS A RANGE OF     1.20 TO     1.40
 7 REPRESENTS A RANGE OF     1.40 TO     1.60
 8 REPRESENTS A RANGE OF     1.60 TO     1.80
 9 REPRESENTS A RANGE OF     1.80 TO     2.00
```


192

```
 RUN
REPRESENTATIONS OF THREE-DIMENSIONALFUNCTIONS


REPRESENTATIONS OF F(X,Y)
FOR X FROM ? TO ?
?-10,10
AND FOR Y FROM ? TO ?
?-10,10
```



```
 X-RANGE IS  -10.00 TO   10.00
Y-RANGE IS  -10.00 TO   10.00
 0 REPRESENTS A RANGE OF     0.00 TO   20.00
 1 REPRESENTS A RANGE OF    20.00 TO   40.00
 2 REPRESENTS A RANGE OF    40.00 TO   60.00
 3 REPRESENTS A RANGE OF    60.00 TO   80.00
 4 REPRESENTS A RANGE OF    80.00 TO  100.00
 5 REPRESENTS A RANGE OF   100.00 TO  120.00
 6 REPRESENTS A RANGE OF   120.00 TO  140.00
 7 REPRESENTS A RANGE OF   140.00 TO  160.00
 8 REPRESENTS A RANGE OF   160.00 TO  180.00
 9 REPRESENTS A RANGE OF   180.00 TO  200.00
```


193

SAMPLE TEST

Computer Math

Name_____
Period_____

TEST ON TOPIC 12

5 Pts. each

True - False

_____ 1. A function definition should be given a lower-numbered line than any program statement that references the function.

_____ 2. DEF FNC(Y) = 2 + FNC(Y) is a valid statement.

_____ 3. A function that has been defined by a DEF statement can be referenced as many times as needed in the rest of the program.

Circle or describe the error in each of these DEF statements.

4. 20 DEF FN(X,Y) = X ↑2 + Y ↑2

5. 20 DEF R(S) = S↑2

6. 20 DEF FNY (A,B,C) = A↑2 + B

Using the function definition statement

10 DEF FNA(X) = X ↑2 - 4*X+3

find these values:

_____ 7. Z = FNA(X) if X = 4

_____ 8. Q = FNA(P) if P = 5

Write a DEF statement for each of the following. Name each function C and use all variables as arguments

_____ 9. (A + B - C) 17

_____ 10. $\sqrt{X \cdot Y}$

Given the function: FNP(W, Q) = (W + Q) * (W - Q) find each of the following:

_____ 11. Z = FNP(4,3)

_____ 12. Z = SQR(FNP(5,3))

_____ 13. Z = FNP(3+4, 4-3)

194

15 Pts.

14. What will be printed when the program is run?

```
100 FOR I = 1 TO 3
110 READ A, B
120 GOSUB 300
130 NEXT I
140 GO TO 999
300 PRINT I; A; B; A + B
310 RETURN
500 DATA 3,5,8
510 DATA 6,-2,1
999 END
```

1	3	5	8
2	8	6	14
3	-2	1	-1

20 Pts.

15. Write a program which will do the following:

1) Define a function $f(X)=3X^2+X-6$
2) Let X take on the values 1,2...10 and evaluate f(X)
3) Print each value of X and f (X).

```
10 DEF FNC(X)=3*X↑2+X-6
20 FOR X = 1 TO 10
30 LET Y = FNC(X)
40 PRINT X, Y
50 NEXT Y
60 END
```

Bonus (6 Pts.)

What is the output?

```
110 LET X=15
120 GOSUB 200
130 PRINT X
140 GO TO 999
200 LET X = X + 1
210 GOSUB 300
220 RETURN
300 LET X = (X+1) * (X+1)↑2
310 RETURN
999 END
```

343

195

ANSWER KEY

Computer Math

Name _____
Period _____

SAMPLE TEST

TEST ON TOPIC 12

5 Pts. each

True - False

T 1. A function definition should be given a lower-numbered line than any program statement that references the function.

F 2. DEF FNC(Y) = 2 + FNC(Y) is a valid statement.

T 3. A function that has been defined by a DEF statement can be referenced as many times as needed in the rest of the program.

Circle or describe the error in each of these DEF statements.

4. 20 DEF FN(X,Y) = X ↑ 2 + Y ↑ 2 — Missing name

5. 20 DEF R(S) = S ↑ 2 — Needs FN

6. 20 DEF FNY (A,B,C) = A ↑ 2 + B — No C value used here

Using the function definition statement

10 DEF FNA(X) = X ↑ 2 - 4*X+3

find these values:

3 7. Z = FNA(X) if X = 4

8 8. Q = FNA(P) if P = 5

Write a DEF statement for each of the following. Name each function C and use all variables as arguments.

DEF FNC (A,B,C)=(A+B-C)/7 9. (A + B - C) 17

DEF FNC(X,Y)= SQR(X*Y) 10. $\sqrt{X \cdot Y}$

Given the function: FNP(W, Q) = (W + Q) * (W - Q) find each of the following:

7 11. Z = FNP(4,3)

4 12. Z = SQR(FNP(5,3))

48 13. Z = FNP(3+4, 4-3)

196

15 Pts.

14. What will be printed when the program is run?

```
100 FOR I = 1 TO 3
110 READ A, B
120 GOSUB 300
130 NEXT I
140 GO TO 999
300 PRINT I; A; B; A + B
310 RETURN
500 DATA 3,5,8
510 DATA 6,-2,1
999 END
```

20 Pts.

15. Write a program which will do the following:

1) Define a function $f(x)=3x^2+x-6$
2) Let X take on the values 1,2...10 and evaluate f(x)
3) Print each value of x and f (x).

Bonus (6 Pts.)

What is the output?

```
110 LET X=15
120 GOSUB 200
130 PRINT X
140 GO TO 999
200 LET X = X + 1
210 GOSUB 300
220 RETURN
300 LET X = (X+1) * (X+1)↑2
310 RETURN
999 END
```

Objective A17a(a) - The student will determine the greatest common factor and least common multiple by using the Euclidean Algorithm.

A17a(b) - The student will determine the 10 smallest Ramanujan numbers.

A17a(c) - The student will find both solutions of a quadratic equation, whether real or complex.

A17a(f) - The student will calculate binomial coefficients.

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE

Objective A17a(a): The Euclidian Algorithm provides an excellent computer method for determining greatest common factors and least common multiples. See Supplement A17a(a). Program A17a(a) of the Program Appendix illustrates the method.

Objective A17a(b): G.H. Hardy remarked to S. Ramanujan, famous Indian number theorist, that 1729, the number of a taxicab he had taken, was rather dull. Ramanujan replied, "No, Hardy! It is a very interesting number. It is the smallest number expressible as the sum of two cubes in two different ways."

In fact, $1729 = 1^3 + 12^3 = 10^3 + 9^3$. Such numbers are here called Ramanujan numbers. Program A17a(b) in the Program Appendix can be used to generate as many as memory and time limitations allow. The problem is discussed in the textbook, Introduction to Computer Applications for Non-Science Students, by William Ralph Bennett, Jr., Prentice-Hall, 1976.

Objective A17a(c): Program A 7a(c) of the Program Appendix shows a correct treatment of complex roots. If for the equation $ax^2+bx+c=0$ the discriminant $D=b^2-4ac$ is negative, then roots of the equation are

$x_1=-\frac{b}{2a}+\frac{i}{2a}\sqrt{-D}$ and $x_2=-\frac{b}{2a}-\frac{i}{2a}\sqrt{-D}$, where $i=\sqrt{-1}$

Objective 17a(f): The Binomial coefficients $\binom{n}{k}$ which is also the number or combinations $\binom{n}{k}$ of n objects taken k at a time, is

$\frac{n!}{k!(n-k)!}$ - Subroutines can be put to good use on this problem. See Program A17a(f) in the Program Appendix.

Supplement A17a(a) - Euclidean Algorithm

Attributed to Euclid is a very clever and efficient procedure to determine the greatest common factor of two numbers. It is best illustrated by example. (The least common multiple can be found at once, when the g.c.f. is known.)

Find the g.c.f. and l.c.f. of 210 and 273.

1. Divide the larger by the smaller.

$$\frac{273}{210} = 1 + \frac{63}{210}$$

2. Drop the integral portion of the quotient, invert the resulting fraction and divide.

$$\frac{210}{63} = 3 + \frac{21}{63}$$

3. Repeat the second step until remainder zero is obtained.

$$\frac{63}{21} = 3 + \frac{0}{21}$$

4. The denominator 21 is the g.c.f.

5. 210/21 = 10. 10 x 273 = 2730, the least common multiple. To appreciate the speed of this method, consider how many steps would be involved in the alternative procedure of reducing each of the other two numbers to factored form and then determining which of the factors are common.

   One further example: Find the g.c.f. and l.c.m. of 1001 and 9991.

$$\frac{9991}{1001} = 9 + \frac{982}{1001}$$

$$\frac{1001}{982} = 1 + \frac{19}{982}$$

$$\frac{982}{19} = 51 + \frac{13}{19}$$

$$\frac{19}{13} = 1 + \frac{6}{19}$$

$$\frac{13}{6} = 2 + \frac{1}{6}$$

$$\frac{6}{1} = 6 + \frac{0}{1}$$

The numbers are relatively prime:

1 is the g.c.f.
9991/1 = 9991. 9991 x 1001 is the l.c.m.

Objective A17a(c): The student will find both solutions of a quadratic equation, whether real or complex.

A17a(d): The student will convert Arabic numerals to Roman numerals.

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Objective A17a(c): Supplement A17a(c) shows an attempted solution of the quadratic equation. It is well written, with the exception that the complex roots are written as $\frac{-B+\sqrt{\lvert B^2-4AC\rvert}}{2A}$ $\frac{-B-\sqrt{\lvert B^2-4AC\rvert}}{2A}$ of course, is not correct. See Program A17a(c) of Program Appendix for a correct solution.	Revise lines 50-100 of this program in order to correct it.
Objective A17a(d): Supplement A17a(d) gives a program to convert any positive integer less than 4000 into the corresponding Roman numeral.	

A program to find both solutions of a quadratic equation, whether real or complex.

```
 10 REM SOLUTIONS OF QUADRATIC EQUATIONS
 20 PRINT "GIVE VALUES OF A, B, AND C (IN ORDER) IN THE EQUATION A*X 2+B*X+C=0."
 30 INPUT A,B,C,
 40 LET D = B↑2-4*A*C
 50 LET X1 = (-B+SQR(ABS(D)))/(2*A)
 60 LET X2 = (-B-SQR(ABS(D)))/(2*A)
 70 IF D<Ø THEN 100
 80 PRINT X1; "AND"; X2; "ARE REAL SOLUTIONS OF THE QUADRATIC EQUATION"; A;
    "*X↑2+"; B; "*X+"; C; "=0."
 90 GO TO 110
100 PRINT X1; "I AND"; X2; "I ARE COMPLEX SOLUTIONS OF THE QUADRATIC EQUATION";
    A; "*X↑2+"; B; "*Y+"; C; "=0."
110 PRINT
120 PRINT "DO YOU NEED SOLUTIONS TO ANY ADDITIONAL QUADRATIC EQUATIONS?  (YES OR NO)"
130 INPUT A$
140 IF A$ = "YES" THEN 20
150 END
```

A program to convert Arabic numerals to Roman numerals.

```
 10 REM CONVERSATION OF ARABIC NUMERALS TO ROMAN NUMERALS.
 20 FOR N = 1 TO 7
 30 READ R$(N)
 40 NEXT N
 50 DATA M, D, C, L, X, V, I
 60 PRINT "THIS PROGRAM CHANGES THE ARABIC NUMERALS (1-3999) TO ROMAN NUMERALS."
 70 PRINT
 80 PRINT "GIVE A POSITIVE INTEGER LESS THAN 4000."
 90 INPUT A
100 IF A > 3999 THEN 140
110 IF A < 1 THEN 140
120 IF A >< INT (A) THEN 140
130 GO TO 160
140 PRINT "YOUR NUMBER MUST BE A POSITIVE INTEGER LESS THAN 4000.  TRY AGAIN."
150 GO TO 90
160 LET Q = INT (A/1000)
170 LET R = A - 1000 * Q
180 IF Q = 0 THEN 220
190 FOR J = 1 to Q
200 PRINT R$(1);
210 NEXT J
220 IF R = 0 THEN 340
230 LET Q = INT (R/100)
240 LET R = R - 100 * Q
250 LET N = 1
260 GO SUB 500
270 LET Q = INT (R/10)
280 LET R = R - 10 * Q
290 LET N = 3
300 GO SUB 500
310 LET Q = R
320 LET N = 5
330 GO SUB 500
340 PRINT "IS THE ROMAN NUMERAL FOR"; A; "."
350 PRINT
360 PRINT "DO YOU HAVE MORE ARABIC NUMERALS TO CONVERT?  (YES OR NO)"
370 INPUT A$
380 IF A$ = "YES" THEN 80
390 STOP
500 REM SUBROUTINE
510 IF Q >< 9 THEN 540
520 PRINT R$(N+2); R$(N);
530 GO TO 640
540 IF Q>< 4 THEN 570
550 PRINT R$(N+2); R$(N+1);
560 GO TO 640
570 IF Q < 5 THEN 600
580 PRINT R$(N+1);
590 LET Q = Q-5
600 IF Q = 0 THEN 640
610 FOR J = 1 TO Q
629 PRINT R$ (N+2);
630 NEXT J
640 IF R = 0 THEN 340
650 RETURN
```

202

See also program A17a(d).1 and A17a(d).2 of the Program Appendix for alternative approaches to this problem.

Objective A17a(g): The student will write a program that will play Tic-Tac-Toe, allowing either the student or the computer to start.

TEACHING SUGGESTIONS

As suggested in the objectives, this is a long, involved problem if the student develops his own logic and writes the program from scratch.

Some teachers may wish to have students modify existing programs and, of course, some students will locate existing programs in any case.

Among existing Tic-Tac-Toe programs written in BASIC are:

a) Solution to Problem 79 of Problems for Computer Solution (Teacher Edition), by Stephen J. Rogowski, Creative Computing Press. (The computer always starts first, despite the manner in which the problem is stated.)

b) Pp. 171-172 of Basic Computer Games, Microcomputer Ed. (1978), David H. Ahl, editor, Workman Publishing, N.Y. (Two programs are listed. In the first, the computer starts. In the second, the player can start if he so chooses.)

STUDENT ACTIVITIES IN THIS GUIDE

a) Practice developing the logic for a 2 x 2 Tic-Tac-Toe game before going to the 3 x 3 case.

b) Use symmetry arguments to argue that in the 3 x 3 game there are only three fundamentally independent opening moves.

c) Study a program which allows only the computer to move first, and modify it to allow the player to move first.

d) Take an already existing program and develop for it graphics appropriate for your computer system (high resolution or color, if available).

e) Take an already existing program and modify it so that the player can sometimes win.

203

ADDITIONAL OBJECTIVES - A

Performance Objectives

A1 - Computer Vocabulary: ASCII/EBDEIC

A15 - The student will write a BASIC program appropriate for the highest level mathematics course the student has taken.

The more advanced student should be able to write programs that will:

A17a(a) - Find the greatest common factor and least common multiple of any two positive integers by using the Euclidean Algorithm.

A17a(b) - Determine the 10 smallest Ramanujan numbers.

A17a(c) - Find both solutions of a quadratic equation, whether real or complex.

A17a(d) - Convert Arabic numerals to Romar numerals.

A17a(f) - Calculate binomial coefficients.

A17a(g) - Play Tic-Tac-Toe, allowing either the student or the computer to start.

204

Objective A15 - The student will write a BASIC program appropriate for the highest level mathematics course the student has taken.

TEACHING SUGGESTIONS	STUDENT ACTIVITIES IN THIS GUIDE
Supplement A15 lists a number of problems broken down into subject areas of Algebra I, Algebra II, Geometry, Trigonometry, Calculus and Analytical Geometry, and Business, Statistics, and Probability. Other good references - (a) Computer Programming in the BASIC Language, by Neal Golden, Harcourt Brace Jovanovich, 1968, has extensive sets of problems from essentially the same categories as those above. (b) Problems for Computer Solution, 2nd ed., by Donald D. Spencer, Hayden Book Co., Inc. (1977). (c) Problems for Computer Solution, by Steven J. Rogowski, Creative Computing Press (1979). (d) Problems for Computer Solution, by Fred Gruenberger and George Jaffray, John Wiley & Sons, Inc. (1965).	Supplement A15

ALGEBRA I APPLICATIONS

NOTEBOOK PROBLEMS

1. Write a program that finds all the real number values of z which make $Cz + D < E$ for C, D, E real numbers.

2. Write a program that will multiply two binomials of the form $(Cx+D)(Ex+F)$.

3. Write a program that will add, subtract, multiply, and divide pairs of complex numbers.

4. Write a program to convert base 5 numbers to base 2 numbers.

COMPUTER PROGRAMS

1. Write a program that will examine the equations of the linear system $Ax + By = C$ and $Dx + Ey = F$ and will print out the following:

   "The lines are parallel and the system is inconsistent."
   "The lines coincide and the system is dependent."
   "The lines intersect in exactly one point. The solution to the system is" (X,Y) "This system is consistent and independent."

2. Write a program that will take the square root of a number. DO NOT use the square root function of the computer or use the power ½.

3. Write a program to find the real number solutions of the equation $|Ax + B| \leq C$ for A, B, C real numbers.

4. Write a program that will convert base 10 numbers to any base up to 12.

5. Write a program to find the reciprocal of a given fraction and will print that reciprocal in fractional form.

6. Write a program that will consider all the possible roots of the quadratic equation $Ax^2 + Bx + C = 0$, given the values A, B, C.

7. Write a program using Cramer's rule to solve a system of two equations written in two unknowns. Have the program print out the equations in standard form and the solution. If the system is inconsistent or dependent have the computer print out this information along with the equations.

ALGEBRA II APPLICATIONS

NOTEBOOK PROBLEMS

1. Write a program that will check the value of $B^2 - 4AC$ of a quadratic equation $Ax^2 + Bx + C = 0$ and will print out the type of solution without solving the equation.

2. Write a program that will find the value for $f(x)$ for the function $f(x) = 3x^2 - 4x + 6$. Let x equal the values 4, 6, 9, 10, 0, 12, and 34

3. Write a program that will use the Factor Theorem to determine whether a given binomial is a factor of a given polynomial.

4. Write a program that will find the roots of a polynomial function, using the location principal.

5. Write a program that will find $f(g(x))$ for $f(x) = 2x - 5$ and $g(x) = x^2 + 4$ where x is in the range -5 to 5

6. Write a program that will read in any matrix and find the transpose of the matrix. If your system has MAT commands DO NOT use them.

7. If your system has MAT commands, write a program that will use these commands to read two matrices, add, subtract, and multiply these matrices (if possible) and will take the inverse and transpose of each of the given matrices.

COMPUTER PROGRAMS

1. Write a program to find the reciprocal of complex numbers.

2. Write a program using Cramer's rule to solve a system of three equations in three unknowns, then expand it to four equations in four unknowns.

3. Write a program that will read two matrices and find their product. DO NOT use the MAT commands.

4. Code and run a program which will (1) find the inverse of a square matrix, (2) find the value of the determinant of a square matrix, and (3) solve for the variables in a system of linear equations (provided that the number of equations is equal to the number of unknowns).

5. Write a program that will use Descartes' rule of signs to assess the nature and possiblilities of the roots of a polynomial equation.

6. Write a program that will determine the possible rational roots of a polynomial equation.

7. Write a program that will find the zeros of a polynomial function.

8. Write a program that will find the quotient and remainder when $2x^4 + 3x^3 - 4x^2 + 5x + 6$ is divided by $x + 3$. Use synthetic division.

GEOMETRY APPLICATIONS

NOTEBOOK PROBLEMS

1. Write a program that will find the measure of each angle of a regular polygon, given the number of sides. Use an INPUT statement. Remember to have instructions prior to the INPUT statement.

2. Write a program that will find the number of sides of a regular polygon given the measure of one of the angles.

3. Write a program that will print a table with appropiate headings for regular polygons with less than ten sides. Use the INPUT statement to enter the number of sides. Have the computer find the size of each interior and exterior angle, the sum of the interior angles, and determine the name of the polygon.

4. Write a program that will read the coordinates of three points and will (1) find the coordinates of the center of the circle determined by these points, (2) find the length of the radius of the circle, and (3) write the equation of the circle.

5. Write a program that will find the area and circumference of a circle, given the radius. Have the printout include the length of the radius, the length of the diameter, the circumference of the circle, and the area of the circle.

6. Write a program that will use the INPUT statement to enter the length of the radius of a circle and the measure of the central angle of the arc, and find the area of the sector and segment of the arc.

7. Write a program that will accept the coordinates of two points on each of two lines and will determine if the lines are parallel perpendicular, or coincide.

8. Given the lengths of three sides of a triangle, write a program that will find the area using Hero's formula.

COMPUTER PROGRAMS

1. Write a program that will use the INPUT statement to enter the measures of three angles and will determine if they are angles in a triangle and if they are, classify the triangle as to acute, obtuse, or right and scalene, isosceles, or equilateral.

2. Write a program that will accept the coordinates of two ordered pairs and will (1) find the midpoints of the segment joining the points, (2) find the length of the segment joining the points, (3) find the slope of the line containing the points, and (4) write the equation of the line containing the points.

3. Modify the program in #3 to write the equation of the perpendicular bisector of the segment joining the points.

Geoemtry (con't)

4. Write a program using the INPUT statement to enter the lengths of three sides of a triangle and determine if the triangle is a right triangle.

5. Write a program that will find the area of a triangle formed when the base is 12 ft. and the altitude varies from 5 ft. to 40 ft. in steps of 5 ft.

6. Write a program that will decide whether three numbers input from the keyboard can be sides of a triangle. If they can, have the program determine whether the triangle is equilateral, isosceles, or scalene. Have the computer print out the lengths and the type of triangle.

7. Write a program that will read the coordinates of three points and will determine whether the points coinside, are collinear, of form a triangle. If a triangle is formed, have the computer print out the perimeter and the area of the triangle.

8. Write a program to find the area of any circle determined by the equation $x^2 + y^2 = r^2$, when the value of r is input.

9. Write a program that will print all the Pythagorean triples with members less than twenty. DO NOT print duplicates.

TRIGONOMETRY APPLICATIONS

NOTEBOOK PROBLEMS

1. Write a program that will print out a table of values for the sine, cosine, and tangent of angles with integral measures between 0 and 90 degrees.

2. Write a program that will convert radian measure to degree measure and degree measure to radian measure. Use an INPUT statement. Be sure your program includes sufficient instructions.

3. Write a program to obtain the polar coordinates of a given function.

COMPUTER PROGRAMS

1. Write a program that will find the sine, cosine, and tangent for any angle in standard position, given the coordinates of a point on the terminal side of the angle. Print out the coordinates and the values of the functions in tabular form.

2. Write a program to find the measures of the angles of a triangle, given the lengths of the three sides.

3. Write a program that will determine the number of triangles formed, given the lengths of two sides and the measure of the angle opposite one of the two sides.

4. Write a program to solve for the missing side and angles of a triangle, given the lengths of two sides and an angle between the two sides.

5. Write a program to solve for the missing sides and angle of a triangle, given the measure of two angles and the side between the angles.

6. Write a program to convert polar coordinates to rectangular coordinates.

7. Write a program that will use the arctangent function to calculate the measure of an angle in standard position, given the coordinates of a point on the terminal side of the angle. Have the measure of the angle printed in degrees.

CALCULUS AND ANALYTICAL GEOMETRY APPLICATIONS

NOTEBOOK PROBLEMS

1. Write a progam that will find the zero's of a given function: $f(x) = x^4 - 3x^2 - 10$.

2. Write a program that will find the relative maximum, relative minimum, and the inflection of a given function, if there is one.

3. Write a program using the trapezoid method to find the area under a given curve.

COMPUTER PROGRAMS

1. Write a program to find the derivative of any given function.

2. Write a program that will find the real roots of an equation, using Newton's method.

3. Write a program that will find the area under a given curve using rectangular approximation and Simpson's method. Use the curve $y = \frac{1}{x}$ for values of x between 1 and 2.

4. Write a program that will find the area bounded by the graphs of $y = x - 6$, $y = 0$, $x = 0$, and $x = 1$.

5. Write a program that will rotate and translate the axis or curve, such that the curve will be centered at the orgin.

BUSINESS, STATISTICS, PROBABILITY APPLICATIONS

NOTEBOOK PROBLEMS

1. Write a program that will find the mean, median, and mode of a given set of data.

2. Write a program that will read in the seven grades for each of six students and will print out the average of the grades and the letter grade earned. Have the computer print this out in tabular form with the names of the students, their numerical grade average, and the letter grade.

3. Write a program for flip four coins 100 times and print the distribution of the outcomes.

COMPUTER PROGRAMS

1. Write a program that will read into the computer six course grades for each of ten students and will print out a report card for each. Assume that the students are all taking the smae subjects.

2. Write a program that will take a set of test scores and calculate the the average score. Have the program then determine how many scores are above average, below average, equal to the average. Have the program and print a frequency table of the scores and the results.

3. Write a program that will store the book records for each teacher in your school. Use a subroutine that will alphabetize the teachers by last name. Print out the Teacher's Name, Number of Books Ordered, Number of Books Received, and Number of Books Due To the Teacher. Have the Source Program and Date of Books in two seperate programs so the MERGE statement may be used.

4. Write a program that will roll a die 100 times and calculate the probalility of rolling a six using the data collected from the 100 rolls.

4. Write a program that will grade a multiple choice examination with 25 questions and will print out (1) the student's score and letter grade, (2) the number of the question missed by the student, the correct answer and the answer the student gave, and (3) page number references in a textbook for the student to review the missed concepts.

5. Write a program that will deal a 5-card hand from a deck of 52 cards to each of 6 players. Be sure that you do not deal the same card to two players.

APPENDIX

213

Program Appendix

1. A17(b) - N factorial from 1 to 15, with column headings
2. A17(c).1 - Prime numbers, elementary method
3. A17(c).2 - Prime numbers, improved method
4. A17(c).3 - Prime numbers by array storage
5. A17(d) - Integral divisors of an integer
6. A17(f).1 - Place values of an integer
7. A17(f).2 - Place values of an integer (alternative method)
8. A17(g) - Extended Fibonacci sequences
9. A17(h).1 - Prime factors of an integer
10. A17(h).2 - Prime factors of a large integer
11. A17(k) - Rounding off numbers
12. A17a(a).1 - Greatest common factor, elementary method
13. A17a(a).2 - Greatest common factor by Euclidean algorighm
14. A17a(b) - Ramanujan numbers
15. A17a(c) - Roots of a quadratic equation
16. A17a(d).1 - Arabic to Roman numerals, restricted range
17. A17a(d).2 - Arabic to Roman numerals, general range
18. A17a(e) - Magic and Latin squares
19. A17a(f) - Binomial coefficients

1. A17(b) - N factorial from 1 to 15, with column headings.
This is similar to Fig. 8.27, p. 253 of textbook, except that it is simpler and uses column headings.

```
10 PRINT " N";TAB( 20);" N! (N FACTORIAL)"
20 PRINT
40 P= 1
50 FOR I= 1TO 15
60 P= P* I
70 PRINT I;TAB( 20);P
80 NEXT I
90 END
READY

RUN

 N                    N! (N FACTORIAL)

 1                    1
 2                    2
 3                    6
 4                    24
 5                    120
 6                    720
 7                    5040
 8                    40320
 9                    362880
 10                   3.6288E+06
 11                   3.99168E+07
 12                   4.79002E+08
 13                   6.22702E+09
 14                   8.71783E+10
 15                   1.30767E+12
```

2. A17(c).1 - Prime numbers, elementary method.
This is similar to Fig. 12.16, p. 375 of textbook. However, note the STEP 2 in line 40, since I will have only odd divisors. It is simpler than the version in the textbook because it does not use a subroutine. It is not very efficient, taking 40 seconds to run on a Compucolor II microcomputer.

```
0 REM  - PRIME NUMBERS, ELEMENTARY METHOD
10 N= 300
20 PRINT 2,3,
30 FOR I= 3TO N STEP 2
40 FOR J= 3TO I/ 2 STEP 2
50 IF I/ J= INT (I/ J)THEN 80
60 NEXT J
70 PRINT I,
80 NEXT I
90 END
```

3. A17(c).2 - Prime numbers, improved method.
This is an improvement over A17(c).1 because of the SQR(I) in line 40. If a number is composite, one factor must always be equal to or less than its square root. Note the addition of 1E-5, since SQR may, because of roundoff, give a result slightly smaller than the true value. Without the 1E-5 numbers which are perfect squares may erroneously be listed as prime. This program runs in about 20 sec. on Compucolor II. Whether or not itis necessary to include 3 in line 20 depends on how looping is handled by your computer when the index is greater than its stated terminal value. (See Supplement A12.5, Topic 5 discussion.)

```
 10 N= 300
 20 PRINT 2.3.
 30 FOR I= 3TO N STEP 2
 35 REM  - TEST WHETHER I IS PRIME
 40 FOR J= 3TO SQR (I)+ 1E- 5STEP 2
 45 REM  - IF I FACTORS THEN ONE FACTOR < SQR(I)
 46 REM  - 1E-5 ADDED BECAUSE SQR(I) MAY NOT BE EXACTLY ACCURATE
 50 IF I/ J= INT (I/ J)THEN 80
 60 NEXT J
 70 PRINT I.
 80 NEXT I
```

4. A17(c).3 - Prime numbers by array storage.
This program stores primes in an array and prints them neatly, 10 per line. For some systems the SI in lines 50 and 75 should be changed to S1. Because of the use of the array this method is intermediate between the one used above and the sieve method of program B17a(1). However the sieve method is much faster, requiring only 4 seconds rather than the 23 needed for the present program. On some systems an END statement must be added to each of the programs on this page.

```
10 DIM A(100)
20 A(1)= 2:A(2)= 3
30 K= 1
40 FOR I= 3TO 299STEP 2
50 SI= SQR (I)+ 1E- 5
55 REM.  CHECK THAT NO SMALLER PRIME DIVIDES I
70 FOR N= 2TO K
75 IF A(N)> SITHEN 100
80 IF I/ A(N)= INT (I/ A(N))THEN 130
90 NEXT N
100 K= K+ 1
110 A(K)= I
130 NEXT I
135 T= 1
140 FOR N= 1TO K
150 PRINT TAB( 5* T);A(N);
155 T= T+ 1:IF T= 11THEN T= 1:PRINT
160 NEXT N
170 PRINT :PRINT
180 PRINT "THERE ARE";K;" PRIMES LESS THAN 300"

READY

RUN

      2    3    5    7   11   13   17   19   23   29
     31   37   41   43   47   53   59   61   67   71
     73   79   83   89   97  101  103  107  109  113
    127  131  137  139  149  151  157  163  167  173
    179  181  191  193  197  199  211  223  227  229
    233  239  241  251  257  263  269  271  277  281
    283  293

THERE ARE 62 PRIMES LESS THAN 300
```

5. A17(d) - Integral divisors of an integer.
This is similar to Fig. 12.14 on p. 374 of the textbook except a subroutine is not used. The objective is simple, and can easily be met during first semester by avoiding use of the subroutine. Note in line 20 that if N is odd, N/2 is automatically interpreted as INT(N/2).

```
10 INPUT N
20 FOR I= 1TO N/ 2
30 IF N/ I= INT (N/ I)THEN PRINT I;
50 NEXT I
60 PRINT N
```

6. A17(f).1 - Place values of an integer.
The method of this program is to pick off the digits of the number, working from right to left. Be sure that students use a loop in their programs rather than writing separate statements to handle each of the six digits.

```
10 REM  - DIGIT VALUES OF A NUMBER
20 INPUT "POSITIVE INTEGER: ";N
30 IF N> = 1E6THEN PRINT "TOO HARD!":GOTO 20
40 R(0)= N
50 FOR I= 1TO 6
60 R(I)= INT (R(I- 1)/ 10)
70 D(I)= R(I- 1)- 10* R(I)
80 NEXT I
90 PRINT "DECOMPOSITION"
100 FOR I= 1TO 6
105 IF D(7- I)= 0THEN 120
110 PRINT D(7- I);" X";TAB( 5);10↑ (6- I)
120 NEXT I
```

7. A17(f).2 - Place values of an integer (alternative method).
This program picks off digits from left to right. Note the use of the INT function and a small increment in line 50 to ensure that, through roundoff in the power function, X is not smaller than expected. Without this addition, errors will sometimes result. An alternative solution to the roundoff problem would be to add line 72, N = INT(N + 0.5) .

```
10 INPUT "CHOOSE AN INTEGER WITH A MAXIMUM OF 6 DIGITS ";N
20 IF N> 999999THEN 10
30 PRINT "THE NUMBER IS";N
40 FOR I= 6TO 1STEP - 1
50 X= 10↑ (I- 1):X= INT (X+ 1E- 5)
60 A(I)= INT (N/ X)
70 N= N- A(I)* X
75 IF A(I)= 0THEN 90
80 PRINT A(I);X;"'S"
90 NEXT I
100 END
RUN

CHOOSE AN INTEGER WITH A MAXIMUM OF 6 DIGITS 124004

THE NUMBER IS 124004
 1 100000'S
 2 10000'S
 4 1000'S
 4 1'S
```

8. A17(g) - extended Fibonacci sequences.
The Fibonacci sequence is shown in Fig. 4.29 on p. 255 of the textbook. Extensions of this sequence are discussed in Supplement A17(g) of Topic 8 in this guide. These extensions are illustrated by the present program.

```
10 REM  EXTENDED FIBONACCI SEQUENCE
15 INPUT "WHICH SEQUENCE? ";K
20 INPUT "NUMBER OF TERMS? ";N
30 DIM F(K)
40 FOR I= 1TO K:F(I)= 1:NEXT I
50 FOR I= 1TO N
55 IF I< = KTHEN 70
58 S= 0
60 FOR J= 1TO K
62 S= S+ F(J)
65 F(J- 1)= F(J)
67 NEXT J:F(K)= S
70 PRINT I,F(K)
80 NEXT I
90 END

READY

RUN

WHICH SEQUENCE? 2

NUMBER OF TERMS? 10

 1  1
 2  1
 3  2
 4  3
 5  5
 6  8
 7  13
 8  21
 9  34
 10  55


READY

WHICH SEQUENCE? 4

NUMBER OF TERMS? 10

 1  1
 2  1
 3  1
 4  1
 5  4
 6  7
 7  13
 8  25
 9  49
 10  94


READY
```

9. A17(h).1 - Prime factors of an integer.
This problem is more complicated than that of finding all factors, since only prime factors are required. The present program is straightforward and adequate for integers up to 500. In line 100 it would be best to add 0.01 or some similar small number to the SQR(I) in case the SQR has been rounded down slightly.

```
10 INPUT "CHOOSE AN INTEGER LESS THAN 500: ";N
20 C= 0
30 IF N> 500THEN 10
40 IF N= 1THEN 180
50 PRINT "PRIME FACTORS OF";N;" ARE: ";
60 IF N= 2OR N= 3THEN 160
70 FOR I= 2TO N/ 2
80 IF N/ I< > INT (N/ I)THEN 150
90 IF I= 2THEN 130
100 FOR D= 2TO SQR (I)
110 IF I/ D= INT (I/ D)THEN 150
120 NEXT D
130 C= C+ 1
140 PRINT I;
150 NEXT I
160 IF C= 0THEN PRINT N:PRINT N;" IS PRIME."
170 GOTO 190
180 PRINT "1 IS NEITHER PRIME NOR COMPOSITE."
190 END

READY

RUN

CHOOSE AN INTEGER LESS THAN 500: 497

PRIME FACTORS OF 497 ARE:  7 71
```

10. A17(h).2 - Prime factors of a large integer.
This program is similar to the previous one, except that an attempt has been made to handle larger numbers efficiently. Thus (line 95) once a prime factor has been found, it is divided out of the original number, making it simpler to search efficiently for larger prime factors. The present program factors 4807 in less than 1 second, whereas the previous one takes 45 sec. on Compucolor II. The present program could be improved by checking both I and N/I for primality.

```
10 INPUT N
20 PRINT "PRIME FACTORS OF ";N
25 IF N/ 2= INT (N/ 2)THEN N= N/ 2:PRINT 2,
35 IF N/ 2= INT (N/ 2)THEN N= N/ 2:GOTO 35
40 IF N/ 3= INT (N/ 3)THEN N= N/ 3:PRINT 3,
50 I= 3
60 REM  - LINE 70 ALLOWS INCLUSION OF N AS PRIME
70 K= N/ I:IF I> KTHEN I= N:GOTO 100
80 REM  - FIND FACTOR OF N
90 IF K< > INT (K)THEN 140
95 N= K
100 FOR J= 3TO INT (SQR (I)+ 1E- 5)STEP 2
110 IF I/ J= INT (I/ J)THEN 140
120 NEXT J
130 IF N= 1THEN END :PRINT I,
140 IF I> = NTHEN END
150 I= I+ 2:GOTO 70

READY
```

11. A17(k) - Rounding off numbers.
This program illustrates how to use the INT function to round off numbers. Note that addition of the 0.5 causes numbers to be rounded up if the previous digit is 5 or more. Without it, the number would just be truncated.

```
10 REM  - COMP MATH A, 17(K): DATA MANIPULATION
20 REM  - INPUT NUMBER IS INCREASED BY 10% AND ROUNDED
30 PRINT "NUMBER TO BE ROUNDED";:INPUT N
40 N0= 1.1* N
50 N1= INT (N+ .5)
60 N2= 100* INT (N/ 100+ .5)
70 N3= .01* INT (N* 100+ .5)
80 PRINT :PRINT N;" ROUNDED TO THE NEAREST"
90 PRINT TAB( 10);"HUNDRED IS";N2
100 PRINT TAB( 10);"INTEGER IS";N1
110 PRINT TAB( 10);"HUNDREDTH IS";N3

READY


RUN

NUMBER TO BE ROUNDED?987.654


 987.654 ROUNDED TO THE NEAREST
          HUNDRED IS 1000
          INTEGER IS 988
          HUNDREDTH IS 987.65
```

12. A17a(a).1 - Greatest common factor, elementary method.
This is a straightforward but inefficient method for finding the greatest common factor of two numbers. It is similar to one found in some textbooks.

```
100 REM  - VERY INEFFICIENT PROGRAM FOR GCF
105 REM  - ALTHOUGH IT IS SIMILAR TO ONE FOUND IN SOME TEXTBOOKS
106 REM  - IT DOESN'T EVEN FIND THE LCM. AND THE
107 REM  - TIME ON MICROCOMPUTER FOR 1024,3125 IS ABOUT 35 SEC.
110 PRINT "WHAT ARE THE VALUES OF THE TWO INTEGERS";
120 INPUT A,B
130 FOR X= 1TO A
140 D= INT (A/ X)
150 E= A- X* D
160 F= INT (B/ X)
170 G= B- X* F
180 IF E< > 0THEN 210
190 IF G< > 0THEN 210
200 Z= X
210 NEXT X
220 PRINT
230 PRINT Z;" IS THE G.C.F. OF";A;" AND";B
240 GOTO 110
```

13. A17a(a).2 - Greatest common factor by Euclidean algorithm.
Supplement A17a(a), under Additional Objectives A of this guide, explains the Euclidean algorithm for finding the gcf of two numbers. The present program uses this algorithm to calculate the gcf and lcm (least common multiple). It is an extremely efficient method, requiring less than 1 second, compared to 35 for the previous program, to find the gcf of 1024 and 3125.

```
LIST


0 REM   GCF-LCM BY THE EUCLIDEAN ALGORITHM
1 REM  - MICROCOMPUTER TIME FOR 1024,3125 IS LESS THAN 1 SEC.
2 REM  - THE LCM AND RELEVANT INFORMATION ARE ALSO INCLUDED.
3 DIM B(15)
5 PRINT TAB( 30);"G.C.F.-L.C.M.":PRINT :PRINT
10 INPUT "ENTER 2 POSITIVE INTEGERS  ";B(0),B(1)
15 REM  - LOCATE THE SMALLER NUMBER
20 IF B(1)> B(0)THEN B= B(0):B(0)= B(1):B(1)= B
25 REM  - THE ALGORITHM
30 I= 0
40 Q= INT (B(I)/ B(I+ 1)):B(I+ 2)= B(I)- Q* B(I+ 1)
50 IF B(I+ 2)< > 0THEN 80
54 PRINT "GCF OF";B(0);" AND";B(1);" IS";B(I+ 1)
56 PRINT B(0);" =";B(I+ 1);" X";B(0)/ B(I+ 1);" AND";B(1);" =";B
(I+ 1);" X";B(1)/ B(I+ 1)
60 PRINT "LCM=";B(0)/ B(I+ 1)* B(1);" =";B(0);" X";B(1)/ B(I+ 1)
;
62 PRINT " =";B(1);" X";B(0)/ B(I+ 1)
65 PRINT :PRINT :INPUT "PRESS RETURN TO CONTINUE";A
70 IF A= 0THEN GOTO 10
75 END
80 I= I+ 1:GOTO 40

READY

RUN

                                G.C.F.-L.C.M.


ENTER 2 POSITIVE INTEGERS  1728,1980

GCF OF 1980 AND 1728 IS 36
 1980 = 36 X 55 AND 1728 = 36 X 48
LCM= 95040 = 1980 X 48 = 1728 X 55


PRESS RETURN TO CONTINUE
```


OSS

14. A17a(b) - Ramanujan numbers.
Ramanujan numbers are discussed under Additional Objectives A17a(b) of this guide. The present program will calculate the first 10 in a machine with at least 16K memory, and more if more memory is available.

```
LIST

 0 REM  RAMANUJAN SUM OF CUBES PROGRAM
 2 PRINT TAB( 20);"RAMANUJAN":PRINT :PRINT
 3 PRINT " A RAMANUJAN NUMBER IS ONE WHICH CAN BE EXPRESSED AS TH
E"
 4 PRINT "SUM OF TWO CUBES IN MORE THAN ONE WAY.":PRINT :PRINT
 5 REM  - TRY N=40 IF YOU HAVE AT LEAST 16K MEMORY.  ALLOW ABOUT
 7 MINUTES
 6 REM  - WITH LESS MEMORY, TRY N=32 TO FIND THE FIRST 5.
 10 INPUT "LARGEST NUMBER TO BE CUBED=";N:PRINT :PRINT
 20 DIM A(N,N),C(N)
 30 REM  - SAVE CUBES OF INTEGERS IN C
 40 FOR I= 1TO N
 50 C(I)= I* I* 1
 60 NEXT I
 70 REM  PUT SUMS OF CUBES IN TRIANGULAR ARRAY
 80 FOR J= 1TO N
 100 FOR K= JTO N
 120 A(J,K)= C(J)+ C(K)
 140 NEXT K,J
 150 REM  SEARCH FOR DUPLICATE ENTRIES
 155 REM  COMPARE A(1,J) WITH A(K,L)
 160 FOR J= 2TO N- 1
 180 FOR I= 2TO J
 200 FOR L= J+ 1TO N
 220 FOR K= 1TO I- 1
 240 IF A(I,J)< > A(K,L)THEN 300
 260 PRINT I;"↑3+";J;"↑3=";A(I,J)
 270 PRINT K;"↑3+";L;"↑3=";A(K,L):PRINT
 300 IF A(K,L)> A(I,J)THEN K= I- 1
 340 NEXT K
 360 IF A(1,L)> A(1,J)THEN L= N
 380 NEXT L
 400 NEXT I,J
 420 PRINT "OTHER RAMANUJAN NUMBERS ARE GREATER THAN";N* N* N

READY
```

15. A17a(c) - Roots of a quadratic equation.
This program calculates real or imaginary (complex) roots of a quadratic equation.

```
LIST

0 PRINT TAB( 20);"QUADRATIC SOLVER":REM   ENTER A=0 TO END PROGR
AM
5 PRINT :PRINT
10 INPUT "TYPE THE VALUES OF A, B, AND C FOR  AX↑2 + BX + C = 0
: ";A,B,C
15 IF A= 0THEN PRINT "PROGRAM ENDS":END
20 D= B* B- 4* A* C
30 IF D> = 0THEN GOSUB 100
40 IF D< 0THEN GOSUB 200
50 GOTO 5
100 R1= - B/ 2/ A:R2= SQR (D)/ 2/ A
110 X1= R1+ R2:X2= R1- R2
120 PRINT :PRINT "THE ROOTS ARE ";X1;"  AND ";X2
130 RETURN
200 D= - D
210 R= - B/ 2/ A
220 I= SQR (D)/ 2/ A
230 PRINT :PRINT "THE ROOTS ARE COMPLEX.  IF (A,B) STANDS FOR A+
IB, THEN"
240 PRINT "THE ROOTS ARE (";R;",";I;") AND (";R;",";- I;")"
250 RETURN

READY

RUN

                    QUADRATIC SOLVER

TYPE THE VALUES OF A, B, AND C FOR  AX↑2 + BX + C = 0 : 1,1,1

THE ROOTS ARE COMPLEX.  IF (A,B) STANDS FOR A+IB, THEN
THE ROOTS ARE (-.5, .866026) AND (-.5,-.866026)

TYPE THE VALUES OF A, B, AND C FOR  AX↑2 + BX + C = 0 : 1,4,1

THE ROOTS ARE -.267949  AND -3.73205

TYPE THE VALUES OF A, B, AND C FOR  AX↑2 + BX + C = 0 : 0,0,0

PROGRAM ENDS

READY
```

16. A17a(d).1 - Arabic to Roman numerals, restricted range. This program converts Arabic numerals less than 100 to the corresponding Roman numerals. A different approach to this problem is listed under Additional Objectives A17a(d) of this guide.

```
LIST


10 PRINT
15 INPUT "CHOOSE A POSITIVE INTEGER < 100 ";N
18 IF N> = 100THEN 15
20 PRINT N;" IN ROMAN NUMERALS IS ";
30 T= INT (N/ 10)
40 ON T+ 1GOTO 140,50,60,70,80,90,100,110,120,130
50 PRINT "X";:GOTO 140
60 PRINT "XX";:GOTO 140
70 PRINT "XXX";:GOTO 140
80 PRINT "XL";:GOTO 140
90 PRINT "L";:GOTO 140
100 PRINT "LX";:GOTO 140
105 GOTO 140
110 PRINT "LXX";:GOTO 140
120 PRINT "LXXX";:GOTO 140
130 PRINT "XC";:GOTO 140
140 U= N- 10* T
150 ON U+ 1GOTO 160,170,180,190,200,210,220,230,240,250
160 PRINT :GOTO 10
170 PRINT "I":GOTO 10
180 PRINT "II":GOTO 10
190 PRINT "III":GOTO 10
200 PRINT "IV":GOTO 10
210 PRINT "V":GOTO 10
220 PRINT "VI":GOTO 10
230 PRINT "VII":GOTO 10
240 PRINT "VIII":GOTO 10
250 PRINT "IX":GOTO 10
260 END

READY
```

17. A17a(d).2 - Arabic to Roman numerals, general range.
The present program is similar to the previous, except that numbers larger than 4000 can be entered, and arrays have been used to simplify the bookkeeping.

```
LIST

0 PRINT TAB( 15);"ARABIC TO ROMAN NUMERAL CONVERSION":PRINT :PRI
NT
5 PRINT "CAUTION:  ROMANS SELDOM USED NUMBERS AS LARGE AS 4000":
PRINT
10 DIM A$(4,9)
20 A$(1,0)= "":A$(1,1)= "M"
30 A$(2,0)= "":A$(2,1)= "C":A$(2,2)= "CC":A$(2,3)= "CCC"
40 A$(2,4)= "CD":A$(2,5)= "D":A$(2,6)= "DC"
50 A$(2,7)= "DCC":A$(2,8)= "DCCC":A$(2,9)= "CM"
60 A$(3,0)= "":A$(3,1)= "X":A$(3,2)= "XX":A$(3,3)= "XXX"
70 A$(3,4)= "XL":A$(3,5)= "L":A$(3,6)= "LX"
80 A$(3,7)= "LXX":A$(3,8)= "LXXX":A$(3,9)= "XC"
90 A$(4,0)= "":A$(4,1)= "I":A$(4,2)= "II":A$(4,3)= "III"
100 A$(4,4)= "IV":A$(4,5)= "V":A$(4,6)= "VI"
110 A$(4,7)= "VII":A$(4,8)= "VIII":A$(4,9)= "IX"
200 PRINT :INPUT "ARABIC NUMERAL ";X
205 PRINT "EQUALS ROMAN NUMERAL ";
210 K= 1000
220 FOR J= 1TO 4
230 N= INT (X/ K)
240 X= X- N* K
250 K= K/ 10
255 REM   IF NUMBERS ARE LIMITED IN SIZE TO LESS THAN 2000, LINE
 250 CAN BE OMITTED.
260 IF J= 1AND N> 1THEN PRINT "M";:N= N- 1:GOTO 260
360 IF N< > 0THEN PRINT A$(J,N);
370 NEXT J:PRINT :PRINT
380 GOTO 200

READY

RUN

                ARABIC TO ROMAN NUMERAL CONVERSION

CAUTION:  ROMANS SELDOM USED NUMBERS AS LARGE AS 4000

ARABIC NUMERAL 1981

EQUALS ROMAN NUMERAL MCMLXXXI


ARABIC NUMERAL 2001

EQUALS ROMAN NUMERAL MMI


ARABIC NUMERAL 444
EQUALS ROMAN NUMERAL CDXLIV

ARABIC NUMERAL 6789

EQUALS ROMAN NUMERAL MMMMMMDCCLXXXIX
```

18. A17a(e) - Magic and Latin squares.
The program of Fig. 11.15 on p. 336 of the textbook calculates row, column, and diagonal sums of a square array. If all sums are the same then the array is a magic square. The present program extends the logic of the textbook program to determine whether or not, in addition, the square is Latin. (The program is continued on the next page. The END statement of line 410 would be a STOP statement for some systems.)

```
LIST


0 REM  - CHECKS FOR MAGIC SQUARE OF SIZE N X N
5 REM  - IN A LATIN SQUARE ONLY N NUMBERS ARE USED.
15 REM  - EACH OCCURS ONCE IN EACH ROW AND ONCE IN EACH COLUMN.
20 PRINT "SIZE OF SQUARE IS IN DATA LINE 320: N=";:READ N:PRINT
N:PRINT
25 DIM L(N,N),S(2* N+ 2)
30 FOR I= 1TO N
35 FOR J= 1TO N:REM  - READ, PRINT ROW BY ROW
40 READ L(I,J):PRINT L(I,J),:NEXT J
45 PRINT :NEXT I:PRINT
50 GOSUB 500
55 FOR J= 1TO N
60 S= 0
70 FOR I= 1TO N
80 S= S+ L(I,J):NEXT I
100 PRINT "COLUMN";J;" SUM IS";S
105 S(J)= S
110 NEXT J:PRINT
120 FOR I= 1TO N:S= 0
140 FOR J= 1TO N
150 S= S+ L(I,J):NEXT J
170 PRINT "ROW";I;" SUM IS";S
175 S(N+ I)= S
180 NEXT I:PRINT :S= 0
200 FOR K= 1TO N
210 S= S+ L(K,K):NEXT K
230 PRINT "MAIN DIAGONAL SUM = ";S
235 S(2* N+ 1)= S
240 S= 0
250 FOR I= 1TO N
260 S= S+ L(I,N+ 1- I)
300 NEXT I
310 PRINT "2ND DIAGONAL SUM =";S
315 S(2* N+ 2)= S
320 DATA 4
322 DATA 4,1,2,3
325 DATA 1,2,3,4
330 DATA 2,3,4,1
335 DATA 3,4,1,2
340 DATA 4,1,2,3
360 PRINT :FOR K= 2TO 2* N+ 2
370 IF S(K)< > S(1)THEN 400
380 NEXT K
390 PRINT "THE ARRAY IS A MAGIC SQUARE OF SUM";S(1)
395 END
400 PRINT "THE ARRAY IS NOT A MAGIC SQUARE."
410 END
```

```
490 REM   CHECK FOR LATIN SQUARE
500 F= 0:FOR I= 1TO N
510 FOR J= 1TO N- 1
520 FOR K= J+ 1TO N
530 IF L(I,J)< > L(I,K)THEN 550
540 PRINT "NOT A LATIN SQUARE.  DUPLICATE NUMBER IN ROW";I:PRINT
 :F= 1
550 NEXT K,J,I
560 FOR J= 1TO N
570 FOR I= 1TO N- 1
575 FOR  K= I+ 1 TO  N
580 IF L(I,J)< > L(K,J)THEN 610
590 PRINT "NOT A LATIN SQUARE.  DUPLICATE NUMBER IN COLUMN";J:PR
INT :F= 1
610 NEXT K,I,J
650 IF F= 1THEN 700
660 REM   CHECK THAT ARRAY CONTAINS ONLY N DISTINCT NUMBERS
670 FOR I= 2TO N
675 FOR J= 1TO N
680 FOR K= 1TO N
685 IF L(I,J)= L(1,K)THEN Q1= Q1+ 1
690 NEXT K,J,I
695 IF Q1= N* (N- 1)THEN PRINT "THE ARRAY IS A LATIN SQUARE.":PR
INT :GOTO 700
697 PRINT "NOT A LATIN SQUARE.  TOO MANY DISTINCT NUMBERS.":PRIN
T
700 RETURN

READY


RUN

SIZE OF SQUARE IS IN DATA LINE 320: N= 4

 1 3 4 2
 4 2 1 3
 2 4 3 1
 3 1 2 4

THE ARRAY IS A LATIN SQUARE.

COLUMN 1 SUM IS 10
COLUMN 2 SUM IS 10
COLUMN 3 SUM IS 10
COLUMN 4 SUM IS 10

ROW 1 SUM IS 10
ROW 2 SUM IS 10
ROW 3 SUM IS 10
ROW 4 SUM IS 10

MAIN DIAGONAL SUM =  10
2ND DIAGONAL SUM = 10

THE ARRAY IS A MAGIC SQUARE OF SUM 10
```



```
READY
```

```
RUN

SIZE OF SQUARE IS IN DATA LINE 320: N= 4

 1 2 3 4
 2 3 4 1
 3 4 1 2
 4 1 2 3

THE ARRAY IS A LATIN SQUARE.

COLUMN 1 SUM IS 10
COLUMN 2 SUM IS 10
COLUMN 3 SUM IS 10
COLUMN 4 SUM IS 10

ROW 1 SUM IS 10
ROW 2 SUM IS 10
ROW 3 SUM IS 10
ROW 4 SUM IS 10

MAIN DIAGONAL SUM =  8
2ND DIAGONAL SUM = 16

THE ARRAY IS NOT A MAGIC SQUARE.


READY
```

```
RUN

SIZE OF SQUARE IS IN DATA LINE 320: N= 4

 16 2 3 13
 5 11 10 8
 9 7 6 12
 4 14 15 1

NOT A LATIN SQUARE.  TOO MANY DISTINCT NUMBERS.

COLUMN 1 SUM IS 34
COLUMN 2 SUM IS 34
COLUMN 3 SUM IS 34
COLUMN 4 SUM IS 34

ROW 1 SUM IS 34
ROW 2 SUM IS 34
ROW 3 SUM IS 34
ROW 4 SUM IS 34

MAIN DIAGONAL SUM =  34
2ND DIAGONAL SUM = 34

THE ARRAY IS A MAGIC SQUARE OF SUM 34
```

19. A17(f) - Binomial coefficients.
This program uses a subroutine to calculate binomial coefficients. Although the subroutine is not essential, it simplifies the programming, since each coefficient requires the calculation of three factorials. The program ends with an OD ERROR (out of data). While there is nothing wrong with terminating a program in this way, it can be avoided by using a flag datum and an IF statement to test whether or not the flag has been read.

```
10 PRINT "COMBINATIONS - BINOMIAL COEFFICIENTS : C(N,K)"
15 PRINT :PRINT
20 REM   LIST VALUES OF N AND K IN DATA STATEMENT
25 REM   THE PROGRAM, AS WRITTEN, WILL END WITH AN 'OUT OF DATA'
 ERROR NOTIFICATION.
30 READ N,K
40 DATA 6,0,6,1,6,2,6,3,6,4,6,5,6,6
50 Z= N:GOSUB 200:Z1= Z
60 Z= K:GOSUB 200:Z2= Z
70 Z= N- K:GOSUB 200:Z3= Z
80 Z= Z1/ Z2/ Z3
90 PRINT "C(";N;",";K;") =";Z
110 GOTO 30
200 P= 1
210 FOR J= 1TO Z
220 P= P* J
230 NEXT J
240 Z= P
250 RETURN

READY

RUN

COMBINATIONS - BINOMIAL COEFFICIENTS : C(N,K)


C( 6, 0) = 1
C( 6, 1) = 6
C( 6, 2) = 15
C( 6, 3) = 20
C( 6, 4) = 15
C( 6, 5) = 6
C( 6, 6) = 1


OD ERROR IN 30
```

20. B3 - Pitfalls of computer arithmetic. (The program is on the next page.) Several pitfalls of computer arithmetic are illustrated by this program and its output shown following the program. Included are roundoff in the square root function and the need for using tolerances; overflow and underflow; roundoff error in the power operation; and roundoff error due to subtraction of nearly equal numbers. To use this program effectively, read the output; then check the program to see how it was generated.

HOUSTON INDEPENDENT SCHOOL DISTRICT
INSTRUCTIONAL MEDIA SERVICES
PROFESSIONAL LIBRARY

BIBLIOGRAPHY: MICROCOMPUTERS

Adams, J. Mack. Social Effects of Computer Use and Misuse. John Wiley & Sons. 1976.

Ahl, David H., ed. 101 Basic Computer Games. Digital Equipment Corp. 1974.

Ahl, David H. Understanding Mathematics and Logic Using Basic Computer Games. Digital Equipment Corp. 1973.

American Association of School Administrators. Instructional Technology and the School Administrator. AASA. 1970.

Andree, Richard V., Josephine P. Andree, and David D. Andree. Computer Programming: Techniques, Analysis, and Mathematics. Prentice-Hall, Inc. Englewood Cliffs, New Jersey. 1973.

Artist and Computer. Ed. by Ruth Leavitt. Harmony Books. 1976.

Boillot, Michel. Understanding Fortran. West Publishing Co., St. Paul, Minn. 1978.

Bushnell, Don D., ed. The Automation of School Information Systems. Dept. of Audio-Visual Instruction of the National Education Association of the United States. 1964.

Barden, William. How to Program Microcomputers. Howard W. Sams & Co. 1977.

Barden, William. Microcomputers for Business Applications. Howard W. Sams & Co. 1979.

Bosworth, Bruce. Programming in BASIC for Business. Science Research Assoc. 1977.

Bushnell, Donald D. The Computer in American Education. Wiley. 1967.

The BYTE Book of Computer Music. Ed. by Christopher P. Morgan. BYTE Books. 1979.

Classroom Computer News. Miliken Publishing Co. Periodical beginning with March, 1981.

Cohen, Daniel. Human Side of Computers. McGraw. 1975.

The Computer in the School: Tutor, Tool, Tutee. Ed by Robert Taylor. Teachers College Press. 1980.

Computer Programming: BASIC for Microcomputers. Educational Activities, Inc. 1979. 5 Filmstrips and cassette tapes.

Computers and Education. Ed. by R.W. Gerard. McGraw-Hill. 1967.

Computers and Public Policy, proceedings of the symposium "Man and the Computer" held at Dartmouth College, November 30, December 1, 2, 1976.

Cox, Nigel S.M. The Computer and the Library. Archon Books. 1967.

Creative Computing. Periodical beginning with 1977.

Data Processing for Education Administration. American Data Processing. 1968.

Desmonde, William Herbert. Computers and Their Uses. Prentice-Hall. 1964.

Doyle, Lauren. Information Retrieval and Processing. Melville. 1975.

Dwyer, Thomas A. Computer Resource Book - Algebra. Houghton Mifflin. 1975.

Education, U.S.A. Computers: New Era for Education? National School Public Relations Association. 1968.

Entelek Incorporated. Computer-Assisted Instruction Guide. 1968.

Farrell, Edmund J. English, Education, and the Electronic Revolution. National Council of Teachers of English. 1967.

Frederick, Franz J. Guide to Microcomputers. Association for Educational Communications and Technology. 1980

Frenzel, Louis E. The Howard W. Sams Crash Course in Microcomputers. Howard W. Sams & Co.

Gruenberger, Fred. Problems for Computer Solution. John Wiley & Sons. 1965.

Hickey, Albert E. Computer-Assisted Instdruction: A Summary of Research in Selected Areas. Entelek. 1975.

Hickey, Albert E. The Logical Basis of Teaching. Entelek Inc. 1964.

Hickey, Albert E. Research Guidelines for Computer-assisted instruction. Entelek. 1974.

Hoppock, Robert. Occupational Information: Where to Get It and How to Use It In Career Counseling, and Career Development. 4th ed. McGraw-Hill. 1976.

Lekan, Helen A. Index to Computer Assisted Instruction. 2nd ed. Sterling Institute. 1970.

Lien, David A. The BASIC Handbook: An Encyclopedia of the BASIC Computer Language. Compsoft Publishing. 1978.

Luce, Thomas G. Genetics With the Computer. Portsmouth, N.H. 1978.

Martin, James. The Wired Society. Prentice-Hall. 1978.

Mathews, Max V. The Technology of Computer Music. M.I.T. Press. 1969.

Moursund, David. Calculators and Elementary Education. Math Learning Center. 1979.

Mullish, Henry. A Business-like Approach to COBOL. Harper & Row Publishers, Inc. New York. 1979.

Orr, William D. Conversational Computers. Wiley. 1968.

Poole, Lon. Accounts Payable & Accounts Receivable CBASIC. Osborne/McGraw-Hill. 1979.

Poole, Lon. General Ledger CBASIC. Osborne/McGraw-Hill. 1979.

Poole, Lon. Payroll With Cost Accounting CBASIC. Osborne/McGraw-Hill. 1979.

Poole, Lon. Some Common BASIC Programs. 3rd ed. Osborne/McGraw-Hill. 1979.

Rader, Richard A. Entelek Computer-Based Physics Lab. Entelek. 1978.

Sage, Edwin R. Problem-solving With the Computer. Entelek, Inc. 1969.

Sippl, Charles J. Computer Dictionary and Handbook. H.W. Sams. 1966.

Soucek, Branko. Computers in Neurobiology and Behavior. John Wiley & Sons. 1976.

Spencer, Donald D. Computers in Society: The Wheres, Why, and Hows of Computer Use. Hayden Book Co., Inc. 1974.

U.S. Department of Health, Education and Welfare, Office of Education. Computer-Based Vocational Guidance Systems. HEW, Office of Education. 1969.

Uttal, William R. Generative Computer-assisted Instruction in Analytic Geometry. Entelek Inc. 1970.

Wessel, Milton R. Freedom's Edge: The Computer Threat to Society. Addison-Wesley Publishing Co. 1974.

Catalog of 16 mm Computer Films

HISD AV Library

1. "Basic Computer Terms," 16 mm, M7560
2. "Career Awareness-Computer Industry," 11 min., 6980
3. "Computers," 11 min. 6065. Discusses components of a computer and library numbers.
4. "Data Processing - Introductory Principles," 14 min., M7189. Discusses use of punched cards.
5. "What is a Computer," 22 min., M5097. Discusses computer functions, terminology, and applications to space travel planning.

Filmstrips Available from the Mathematics Resource Center

1. Working as a Computer Programmer 4489.018LA
2. How a Computer Solves a Problem 4522.018DG
3. Different Numeration Systems 4522.018G1N
4. An Invitation to Computers 4529.019BFA

Also available at some schools:

5. How Computers Work, Encyclopedia Britannica Educational Corp., 1972